Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп’ютерної інженерії та управління
Кафедра Комп’ютерних інтелектуальних технологій та систем

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Аналіз поведінки користувача
під час проведення е-тестів

Виконав:
студент 2 курсу, групи КІТм-20-1
Татарников А.О.

Спеціальність 123 Комп’ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп’ютерні інтелектуальні технології
Керівник проф. Аксак Н.Г.

Допускається до захисту _____
(підпис)

Зав. кафедри _____ проф. Руденко О.Г.
(підпис)

2021 р.

Харківський національний університет радіоелектроніки

Факультет	Комп’ютерної інженерії та управління
Кафедра	Комп’ютерних інтелектуальних технологій та систем
Рівень вищої освіти	другий (магістерський)
Спеціальність	123 – Комп’ютерна інженерія
Тип програми	освітньо-професійна
Освітня програма	Комп’ютерні інтелектуальні технології

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)

«____» _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Татарникову Андрію Олександровичу
(прізвище, ініціали)

1. Тема роботи (проекту) Аналіз поведінки користувача під час проведення е-тестів

затверджена наказом по університету від " 08 " листопада 2021 р. № 1666Ст

2. Термін подання студентом роботи до екзаменаційної комісії 10 грудня 2021 р.

3. Вхідні дані до роботи

1. Документація мови програмування C# та платформи .NET.
2. Документація технології розробки інтерфейсу застосунків Windows Forms.
3. Інтегроване середовище Visual Studio 2015 Community.
4. Документація CMS WordPress.
5. Документація мови програмування PHP.
6. Платформа Node.js.

4. Перелік питань, що потрібно опрацювати в роботі

1. Аналіз предметної області.
2. Аналіз використовуваних технологій.
3. Програмна реалізація клієнтської та серверної частин.
4. Інструкція користувача.
5. Висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів)

Слайд-презентація – 15 слайдів

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Видача та узгодження теми проекту	08.11.2021	
2	Огляд стану проблеми та постановка задачі	11.11 – 13.11	
3	Аналіз літератури за напрямком магістерської	13.11 – 20.11	
	роботи		
4	Аналіз аналогів, за напрямком розроблюваного	21.11 – 23.11	
	застосунку	24.11 – 26.11	
5	Вибір методів рішення для реалізації та їхнє		
	обґрунтування	26.11 – 27.12	
6	Розробка та тестування клієнтської	28.11 – 29.12	
	частини за стосунку		
7	Розробка та тестування веб-застосунку	29.11 – 30.12	
8	Оформлення пояснювальної записки	01.12 – 07.12	
9	Оформлення графічної частини	08.12 – 10.12	
10	Перевірка виконаного проекту керівником	10.12.2021	
11	Захист проекту	16.12.2021 – 17.12.2021	

Дата видачі завдання 08 листопада 2021 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Аксак Н.Г.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 130 с., 34 рис., 6 табл., 2 дод., 52 джерел.

Метою кваліфікаційної роботи є створення системи спостереження за поведінкою студентів під час проведення е-тестів. Розроблена система дозволяє в режимі реального часу відстежувати порушення під час проведення тестування. Об'єктом дослідження є системи моніторингу за е-тестуванням. Предмет дослідження: використання систем моніторингу як спосіб стеження якості проведення е-тестування.

У ході виконання кваліфікаційної роботи був проведений аналіз великої кількості існуючих програмних рішень, як спеціалізованих так і програмних. в ході дослідження були виявлені основні переваги та недоліки даних застосунків. На основі проведеного аналізу існуючих рішень були сформульовані головні вимоги до функціоналу та принципи роботи за стосунку що розробляється.

Для реалізації клієнтської частини використана мова програмування C# на платформі .NET. Для серверної частини використана CMS WordPress. Інтерфейс користувача веб-застосунку та організація взаємодії з ним реалізується за допомогою основних можливостей HTML, CSS, JavaScript, PHP та C#.

ЕЛЕКТРОНЕ ТЕСТУВАННЯ, КЕЙЛОГЕР, РОЗПІЗНАВАННЯ ОБЛИЧЧЯ, КЛІЄНТ-СЕРВЕРНА АРХІТЕКТУРА, ВЕБ-КАМЕРА

ABSTRACT

Master’s thesis: 130 pages, 34 figures, 6 tables, 2 appendices, 52 sources.

The purpose of the certification work is to create a system for monitoring the behavior of students during e-tests. The developed system allows to track violations during testing in real time.

In the course of attestation work, an analysis of a large number of existing software solutions, both specialized and software, was conducted. the study identified the main advantages and disadvantages of these applications. Based on the analysis of existing solutions, the main requirements for the functionality and principles of work in the relationship being developed were formulated.

The C # programming language on the .NET platform was used to implement the client part. CMS WordPress is used for the server part. The user interface of the web application and the organization of interaction with it is implemented using the basic features of HTML, CSS, JavaScript, PHP and C #.

ELECTRONIC TESTING, KEYLOGER, FACE RECOGNITION, CLIENT-SERVER ARCHITECTURE, WEBCAM

АНОТАЦІЯ

Татарников А.О. Аналіз поведінки користувача під час проведення е-тестів. – Магістерська кваліфікаційна робота.

У магістерській науковій роботі вирішено актуальну науково-прикладну проблему створення спеціалізованої клієнт-серверної системи спостереження та аналізу поведінки студентів під час проведення е-тестування, яка дозволяє максимально ефективно використовувати ресурси шляхом розробки та реалізації методів обробки та аналізу зібраних за час спостереження даних з використанням можливостей комп’ютерного зору.

Об'єктом дослідження цієї роботи є системи моніторингу за е-тестуванням. **Предметом дослідження**: використання систем моніторингу як способу стеження якості проведення е-тестування.

Метою кваліфікаційної роботи є створення системи спостереження за поведінкою студентів, під час проведення е-тестів. Для цього було проведено дослідження систем контролю за процесом тестування. Кожна з досліджуваних систем має в своїй основі унікальний технічний функіонал, завдяки якому вона і стала популярною. Саме тому, щоб розробити систему більш високого рівня ніж аналогічні, було проаналізовано широкий асортимет програмних застосунків.

У роботі виконано системний аналіз існуючих технологій, систем, та програмних за стосунків моніторингу, а саме: AnyDesk, Chrome Remote Desktop, Spyera, DameWare, AeroAdmin, UltraVNC, Віддалений доступ RMS, Remote Utilities, Microsoft Remote Desktop, Proctortrack, Exam Monitor, ProctorEdu, ExamCookie, Екзамус.

Дослідження проводилися на підставі зібраних про програми даних та практичного тестування окремих їх функцій. В процесі дослідження були виявлені найкращі функціональні елементи.

Пропонується комплекс взаємопов’язаних блоків моніторингу та стеження, який включає в себе такі основні можливості

Відстеження активних URL – ця функція дає можливість відстежувати в доступних для операційної системи Windows в найпоширеніших типах браузерів такі параметри як: розпізнавання активного типу браузера, його назва та вміст рядка адреси.

Стеження в режимі реального часу буд-яких змін параметрів (ширини та висоти) активного вікна. Для виявлення порушень виконується порівняння еталонних параметрів (отримуються одразу після запуску) через заданий в налаштуваннях інтервал. В разі виявлення змін, отримані нові параметри ширини та висоти вікна записуються та відправляється повідомлення на сервер.

Зчитування списку активних процесів на комп’ютерів тестованого може виконуватися як з певним інтервалом так і по команді проктора. Можливість виявляти та ознайомлюватися зі списком активних в процесі тестування процесів, відкриває можливість виявляти будь-які сторонні програми та утиліти на робочому ПК студента.

Отримання переліку мережевих підключень – дана функція дозволяє захиститися від можливого проходження тесту за допомогою систем віддаленого управління.

Автоматичне зчитування місту буфера обміну та ведення логу натиснутих клавіш (за винятком клавіш миші). Дана функція дозволяє відстежити та оперативно запобігти будь-яким спробам з боку тестованого знайти відповіді на питання за допомогою сторонніх програмам або в Інтернеті.

Створення знімків робочої області екрану комп’ютера та фотознімків за допомогою веб камери ПК. В процесі здачі КТ проктор може увімкнути функцію автоматичного створення знімку екрану при зміні розмірів активного вікна програми, та в разі необхідності знімки можна створювати по команді з серверу.

Розпізнавання обличчя тестованого та стеження за його положенням відносно веб-камери. Дана функція в режимі реального часу детектує особу студента, щоб запобігти можливій заміні в процесі іншою людиною. Та стежити за тим щоб студент не відволікався від процесу проходження тесту.

Серверна частина системи моніторингу за е-тестуванням повинна бути розміщена на окремій машині, без доступу до неї сторонніх особі, для роботи з клієнтами їй необхідне хороше з'єднання з мережею Internet, або налаштована локальна мережа.

Адаптивна система налаштувань кожного елементу контролю. Включає в себе повністю варіативну зміну інтервалу передачі даних від клієнта на сервер і відправку повідомлень від серверу на підконтрольні комп’ютери. Точне налаштування інтервалі передачі даних дозволяє проктору більш точно налаштувати клієнтську програму в залежності від ситуації та важливості тесту. Функція відправки повідомлень з серверу застосовується в ситуаціях, коли проктору необхідно надіслати повідомлення або попередження конкретному студенту.

Забезпечення роботи серверу на пристроях з невеликою роздільною здатністю екрану. Забезпечення такої можливості в подальшому може дати змогу екзаменатору виконувати налаштування та стежити за процесом тестування з будь-якого пристрою віддалено.

Вивантаження інформації в окремий файл. Можливість, що дозволяє, в разі необхідності більш детального ознайомлення, створити спеціальний файл з даними конкретної таблиці або всієї БД, в форматі .csv, для подальшого ознайомлення студентом, або викладачем.

В роботі проведено обґрунтування кожної з запропонованих функцій стеження. Удосконалено взаємодію між клієнтами системи та проктором, що дозволяє більш оперативно стежити за поведінкою тестованих та швидше виявляти будь-які спроби списування з боку учнів. Запропонований метод стеження може працювати в двох основних режимах:

Повністю автономно. В процесі тестування система стеження збирає дані про поведінку студентів та відправляє результати на віддалений сервер для подальшої обробки та перевірки проктором.

Під наглядом проктора. Процес проходження е-тетстування знаходиться повністю під наглядом проктора, який завдяки використанню централізованого

серверного застосунку може в режимі реального часу стежити за процесом тестування та в разі необхідності відправляти попередження тестованим.

Практичне значення отриманих результатів полягає у розробці:

– структури базових модулів клієнт-серверного програмного застосунку прихованого спостереження за поведінкою студентів на основі запропонованих функціональних модулів та систем;

– структури спеціалізованих модулів взаємодії між проктором та тестованими, що дозволяє проводити контроль за тестом віддалено;

– структури взаємодії між клієнтськими машинами та централізованим сервером, що дозволяє досягнути максимальної швидкості обробки та передачі даних;

Запропоновані методи та моделі доведені до рівня практичної реалізації:

– забезпечення доступу до всіх основних компонентів та системних модулів ПК студента дозволило збирати широкий список даних про його поведінку під час проходження ним е-тестування;

– принцип побудови адаптивної клієнт-серверної системи та методи взаємодії між її основними елементами дали можливість налаштовувати роботу кожного окремого модулю в залежності від конкретної ситуації;

– розроблена архітектура взаємодії між клієнтами та сервером дозволила збирати та оброблювати дані, навіть при відключенні Інтернету;

Поставлені в роботі задачі виконані в повному обсязі. Подальший розвиток проекту передбачає реалізацію більш надійної системи синхронізації даних між клієнтською та серверною БД, оптимізацію роботи з користувачами на стороні сервера, підключення до системи комп’ютерного зору можливості розпізнавати емоції тестованого.

Ключові слова: електронне тестування, кейлогер, розпізнавання обличчя, клієнт-серверна архітектура, веб-камера.

Публікації здобувача за темою роботи:

1. Татарников А.О. Використання технології комп’ютерного зору як методу контролю під час онлайн тестування. *Пріоритетні напрямки та вектори розвитку світової науки:* матеріали І міжнародної студентської наукової конференції., м. Миколаїв, 21 травня 2021. Вінниця. С. 15–16.

2. Іващенко Г.С., Татарников А.О. Організація взаємодії та синхронізації даних між sqlite і mysql. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління:*. матеріали десятої міжнародної науково-технічної конференції., м.Харків, 9-10 квітня 2020. Харків. С. 70.

3. Татарников А.О. Завдання інтеграції алгоритмів комп'ютерного зору в системах контролю успішності учнів. *Проблеми та перспективи реалізації та впровадження міждисциплінарних наукових досягнень.* матеріали ІІ міжнародної наукової конференції., м.Київ, 27 серпня 2021. Київ. С. 117–118.

4. Татарников А.О., Іващенко Г.С. Організація клієнт-серверного взаємодія в системі моніторингу проходження тестування учнів. *Радіоелектроніка та молодь у XXI столітті*: матеріали XXIV міжнародного молодіжного форуму., м. Харків, 7-9 квітня 2020. Харків, 2020. С. 25–26.

5. Татарников А.О. Проблема організації захисту даних при використанні протоколу websocket у клієнт-серверній архітектурі. *Об’єднані наукою: перспективи міждисциплінарних досліджень:* матеріали VII Всеукраїнської науково-практичної конференції студентів, аспірантів та молодих вчених., м. Київ. 12-13 листопада 2020. Київ. С. 34–35.

6. Татарников А.О. Використання технології комп’ютерного зору при проведенні онлайн тестування. *Інформаційні технології в сучасному світі: дослідження молодих вчених:* матеріали міжнародної науково-практичної конференції молодих учених, аспірантів та студентів., м. Харків. 18-19 березня 2021. Харків. С. 58.

7. Татарников А.О. Дослідження методів обробки та класифікації емоцій

людини при розробці систем комп’ютерного зору. *Теорія і практика сучасної науки*: матеріали II міжнародної науково-теоретичної конференції., м. Краків. 12 листопада 2021. Краків. С. 91–92.

8. Татарников А.О. Використання технології розпізнавання емоцій людини на основі штучних нейронних мереж в освіті. *Цифровізація науки та сучасні тренди її розвитку:* матеріали II міжнародної студентської конференції., м. Миргород 5 листопада 2021. Миргород. С. 86–87.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД – база даних

КТ – комп’ютерне тестування

ОС – операційна система

ПК – персональний комп’ютер

СКБД – система керування базами даних

API – Application Programming Interface

CMS – Система Управління Вмістом (англ. Content Management System)

HTTP – Протокол Передачі Гіпертексту (англ. HyperText Transfer Protocol)

LMS – Learning Management System

UI – User Interface

URL – Uniform Resource Locator

WPF – Windows Presentation Foundation

ВСТУП

У зв'язку з переведенням навчального процесу на принципово нові навчальні плани істотно зростає роль самостійної роботи студентів, яка стала основною формою отримання знань. В таких умовах виникає гостра необхідність інтеграції спеціалізованих форм контролю навчальної роботи студентів, і автоматизації навчального процесу в цілому. Крім того системи королю за е-тестуванням є дуже перспективними в плані впровадження у систему освіти, можуть мати великий ефект на економіку, та дозволити перевести систематичне засвоєння теоретико-практичного матеріалу студентами на повністю новий рівень.

Однак як і всі новітні технології, підготовка матеріалів для тестів вимагає серйозної підготовки. Це пов'язано з тим, що оцінка завдань і контрольних тестів для великих обсягів студентів, займає значну кількість часу, який можна було використати для взаємодії викладача зі студентами, підготовки до занять або роботи над професійним розвитком вчителя.

Проте спочатку викладачу необхідно оволодіти суворими правилами оформлення, адже неохайно розроблені, тести, в ході розробки яких не було проведено процес апробації, можуть в підсумку надати лише помилкові результати. Що в свою чергу знижує об’єктивність самого тестування як елементу контролю [1].

Однак незважаючи на всі переваги процес тестування все ще являється вразливим до можливих зловживань з боку студентів – використання сторонніх засобів, допомога інших учнів і т.д. У зв'язку із кардинальним збільшенням потоків студентів в навчальних закладах і відповідним зростанням кількості учнів, які одночасно проходять тестування, пропорційно зростає і навантаження на викладача, таким чином, знижується об’єктивність самого тестування [1].

Дана проблема може вирішуватися за допомогою вже розроблених методик онлайн контролю, в яких використовуються принципи системи управління курсами Moodle [2]. В основі даної системи лежить організація взаємодії між викладачем і учнями в режимі онлайн, а також підтримка очного навчання. Так і за допомогою стороннього програмного забезпечення.

До можливостей таких програм можна віднести:

- стеження за активними URL браузерів;
- відстеження та легування мережевих підключень;
- логування підозрілих процесів;
- стеження за вмістом буферу клавіатури;
- створення знімку робочої області екрану комп’ютера.
- запис робочого столу в режимі реального часу
- одночасна робота з декількома віддаленими комп’ютерами
- створення знімків робочої області екрану
- відправка повідомлень між учасниками робочої сесії

Це забезпечує універсальність в роботі з програмним застосунком та відкриває можливість перед викладачами впроваджувати тестування у будь-яку навчальну дисципліну на будь-якому етапі процесу навчання.

Метою роботи є дослідження систем контролю за процесом тестування, пошук найкращих функціональних рішень та розробка програмного застосунку відповідно до необхідної функціональності.

Об'єктом дослідження цієї роботи є системи моніторингу за е-тестуванням.

Предметом дослідження: використання систем моніторингу як способу стеження якості проведення е-тестування.

1 АКТУАЛЬНІСТЬ ПРОБЛЕМИ ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність задачі

На сьогоднішній день зростає необхідність та актуальність комп’ютерного тестування як засобу контролю успішності засвоєння студентами навчальних матеріалів. Це дозволяє прискорити процес контролю та забезпечує його об'єктивність. Однак процес тестування вразливий до зловживань з боку учнів – використання сторонніх засобів, допомога інших учнів і т.д. У зв'язку зі збільшенням кількості студентів в навчальних закладах і відповідним зростанням чисельності потоків учнів, які проходять тестування, зростає навантаження на викладача і, таким чином, може знизитися об’єктивність самого тестування.

Рішенням даної проблеми може стати використання спеціалізованої, комбінованої системи контролю. В такій системі будуть поєднуватися як основні елементи звичайних систем моніторингу, так і переваги систем, що побудовані на основі комп'ютерного зору, наприклад FindFace [5,6]. Дана система може використовуватися для відстеження поведінки студентів, при цьому працювати одночасно з тисячами камер в режимі реального часу. Та здійснювати ідентифікацію, авторизацію, розпізнавання особистості студента, та детектування сторонніх особі, які можуть потрапити у поле зору камер [5]. В разі виявлення будь-якої сторонньої активності як на комп’ютері так і відеокамерою, система в автоматичному режимі буде проводити аналіз даних з усіх доступних елементів контролю. Зібрані дані будуть відправлятися, та зберігатися на спеціальному захищеному сервері, і будуть міститися там до декількох місяців. Під час самого онлайн-тестування, проктор або викладач можуть самостійно стежити за процесом та виявляти будь які порушення. А в разі необхідності ознайомитися з результатами стеження після тестування [5,6].

1.2 Огляд літературних джерел

В умовах сучасного інформаційного суспільства перед освітою виникає глобальна проблема – збільшення кількості та необхідність підвищення якості навчальної інформації. Це вимагає від викладача постійного контролю за сприйняттям нової інформації, уміннями та навичками учнів, що потребує чимало часу [3-6]. Тому у педагогічних інноваціях і виник новий напрямок — комп'ютерне тестування. У якому оцінювання рівня знань студентів буде здійснюватися за допомогою комп’ютерного тестування (КТ).

Процес тестування може проводитись у різних форматах, кожен з них має свою унікальну технологію створення тестового питання. Виділяють три основні форми.

До першої форми відносяться готові тести. Це стандартизовані, призначені для поточного контролю знань учнів. Для роботи використовуються завчасно підготовлені завдання та варіанти відповідей до них.

Друга форма КТ в основі роботи передбачає автоматизовану генерацію варіантів відповідей окремо до кожного варіанту відповіді. Варіанти відповідей можуть створюватися безпосередньо як перед іспитом так і безпосередньо під час його проведення. Такий підхід стає можливим завдяки використанню спеціальних бланків каліброваних тестових завдань.

Комп’ютерне тестування третьої форми являє собою спеціалізований адаптивний тест. В основі роботи лежить міркування про те, що тестованому буде марно давати завдання з якими він напевно впорається правильно, або гарантовано не впорається через високу складність питання. Такий підхід дозволить оптимізувати складність завдань відповідно до рівня підготовленості кожного учня та скоротити за рахунок виключення частини завдань довжину тесту.

Проведення КТ дозволяє отримати об'єктивні оцінки рівня знань, умінь, навичок студентів, виявити прогалини в підготовці та полегшує

роботу екзаменатора при оцінювані результатів тестів. Після впровадження перших типів КТ було відзначено позитивне відношення до процедури КТ, яке проявлялося у певних змінах: зросла активність роботи студентів на заняттях, з'явився дух змагання та бажання успішно скласти тести, посилився інтерес до самостійної підготовки. Це можна пояснити такими факторами, як можливість отримання миттєвого результату, виключення упередженого ставлення, швидкість проходження та розуміння неминучості контролю [5,6].

Отримання миттєвого результату, у присутності студента, дозволяє запобігти сумнівів у об’єктивності отриманого результату. Важливим є виключення упередженого ставлення в оцінці знань. Причини для якого можуть бути різні, наприклад, студент може бути впевнений в упередженому до нього відношенні, або просто робити вигляд, що таке ставлення має місце. Тестування за допомогою КТ практично виключає можливість такого ставлення, особливо якщо тестований знає, що питання тесту обираються програмою випадково. Тому результат тестування трактується не як вираження ставлення викладача, а як необхідність краще вчитися.

Простота використання і швидкість виконання тестів. Даний фактор створює ілюзію простоти та доступності матеріалу, а також легкості самого процесу навчання, що являє собою потужний рушійний стимул [3-6].

Неминучість контролю. При проведенні звичайного заняття контроль є вибірковим та поверхневим. При проведенні комп'ютерного тестування, проводиться контроль кожного учня з усіх питань певної дисципліни. Це мобілізує учнів на більш ретельну підготовку до заняття [3-6].

Проте при всіх перевагах комп’ютерного тестування як засобу контролю всі можливі негативні наслідки такого підходу, через новизну технології, досі не виявлені повною мірою.

Так наприклад вибір використання КТ повинен ґрунтуватися на серйозних передумовах, ніж просто зменшення навантаження на лекторів та екзаменаторів. Через такий підхід може породжуватися безліч проблем, та може статися так що тестовані можуть опинитися в нерівних умовах. Саме

тому використовувати КТ необхідно саме тоді, коли стоїть гостра потреба у неможливості використання бланкових тестів.

1.3 Використання клієнт-серверної архітектури для забезпечення контролю за процесом проведення комп’ютерного тестування

Основна задача архітектури Клієнт-Сервер являє собою поділ всіх процесів, запитів та функцій на різних комп’ютерах у мережі, кожен з яких виконує свої завдання незалежно від інших. На сьогоднішній день прийнято розділяти три основні класи: одно, до та трирівнева архітектура [7].

Сервер виступає в якості персональної чи віртуальної ЕОМ з розподіленими пам'яттю, обробленням даних, комунікаційними засобами та засобами управління периферійним обладнанням (рисунок 1.1).

Рисунок 1.1 – Загальна схема архітектури клієнт-сервер

Зазвичай у якості серверу застосовують потужні ЕОМ, що мають великий дисковий простір і швидкодійні процесори. Основна роль серверу полягає в управлінні клієнтами, які спільно користуються ресурсами системи в заданий момент часу [7,8].

Клієнт – це робоча станція, що взаємодіє з користувачем, яка здатна виконувати потрібні обчислення та забезпечує підключення до обчислювальних ресурсів та БД, засобів обробки інформації, а також засобів організації інтерфейсів. У якості клієнта може бути використаний будь-який

персональний комп'ютер (ПК), завдяки використанню для обробки та роботи з отриманими даними ресурсів більш потужного серверу.

Введення-виведення до бази даних основане не на фізичному розподілі даних, а на логічному, тобто сервер відправляє клієнтам не повну копію бази даних, а тільки логічно необхідні порції. Завдяки цьому скорочується трафік мережі, забезпечується цілісність даних та забезпечується їх безперервне збереження [7,8].

Однорівнева архітектурі всі програми розосереджені серед всід робочих станцій, які в процесі роботи звертаються безпосередньо до одного (спільного) сервера, ресурсів та бази даних [7,8] (рисунок 1.2).

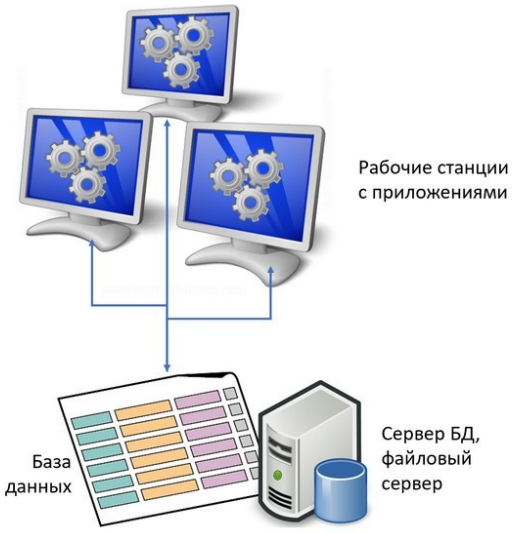

Рисунок 1.2 – Загальна схема одноуровневої архітектури клієнт-сервер

При цьому сервер не виконує жодних фунцій, а лише надає дані на запити від користувача.

В дворівневій архітектурі клієнт-сервер виконуючі програми зосереджені на сервері додатків (Application Server) (рисунок 1.3). А на робочих ПК знаходяться програми-клієнти, завданням яких є надання користувачам інтерфейсу для роботи з за стосунками на сервері [7,8].

Дворівнева архітектура простіша, так як всі запити обслуговуються

одним сервером, але саме через це вона менш надійна і висуває підвищені вимоги до продуктивності сервера [7,8].

В залежності від необхідних функцій розрізняють такі типи моделей:

модель тонкого клієнта, в рамках якої вся логіка застосунку та управління даними зосереджена на сервері. Клієнтська програма забезпечує тільки функції рівня представлення.

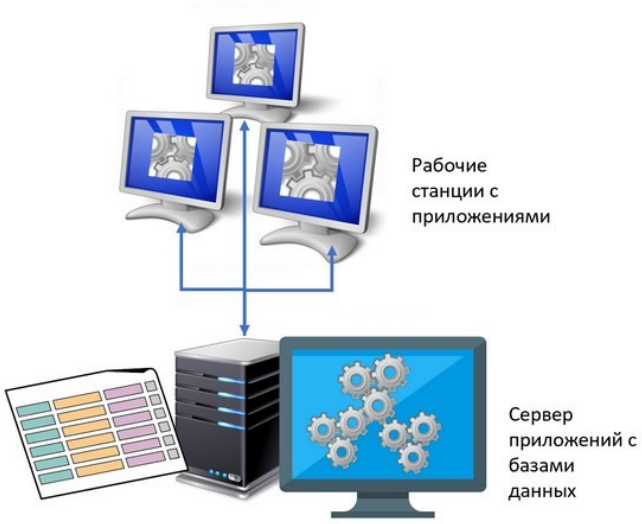

Рисунок 1.3 – Загальна схема дворівневої архітектури клієнт-сервер

Модель товстого клієнта, в якій сервер тільки керує даними, а обробка інформації та інтерфейс користувача зосереджені на стороні клієнта. Товстими клієнтами часто також називають пристрої з обмеженою потужністю: кишенькові комп’ютери, мобільні телефони та ін [7,8].

Трирівнева архітектура являє собою удоскалену модель дворівневої, але завдяки тому, що функції розподілені між серверами другого і третього рівня, ця архітектура проявляє:

– високий ступінь гнучкості і масштабованості.

– високу безпеку (тому що захист можна визначити для кожного сервісу або рівня).

– високу продуктивність (тому що завдання розподілені між

серверами) [6,7].

При використанні трирівневої архітектури використовується розділення на окремі рівні серверу баз даних, файлового менеджера, та ін (рисунок 1.4).

Рисунок 1.4 – Загальна схема трирівневої архітектури клієнт-сервер

Програми проміжного рівня можуть функціювати під управлінням спеціальних серверів застосунків, але запуск таких програм може здійснюватися і під управлінням звичайного веб-сервера. В даному випадку результати роботи оброблюються на рівні серверу застосунків. Внаслідок цього, підвищується гнучкість роботи та продуктивність [7,8].

1.4 Існуючі програмні рішення

Екзамус – програмний за стосунок створений для попередження та запобігання можливих спроб списування з боку студентів під час онлайн-

тестування. Перед початком тестування учень повинен пройти спеціальну процедуру ідентифікації по фотографії, і тільки після цього може приступити до проходження тестування [9].

Головним недоліком даного за стосунку є автономність процесу контролю. Будь-яке задетектоване опускання очей під час процесу тестування програмою можуть розпізнатися як спроба списування [9].

ExamCookie – програмний за стосунок який може в режимі реального часу контролювати активність студентів на комп’ютері (рисунок 1.5).

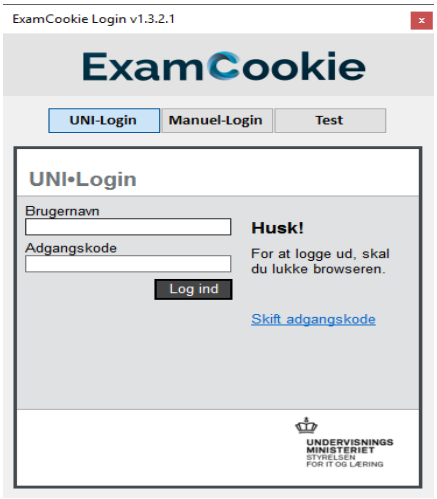

Рисунок 1.5 – Головне вікно програми

Для роботи ExamCookie проводить процес збереження та відстеження будь-якої активності на ПК тестованого.

Головним недоліком цієї програми є те, що учень знає про моніторинг процесу тестування та може втручатися в роботу самої системи, наприклад ввести дані іншої особи [10,11].

ProctorEdu – спеціалізована система прокторингу призначена для онлайн-спостереження за поведінкою студентів під час проходження ними онлайн тестування (рисунок 1.6). Дана система може бути підключена до необхідної платформи тестування (ПК, телефон, планшет), як у автоматичному режимі, так і з безпосередньою участю проктора [12].

Серед основних можливостей застосунку можна виділити:

- скоринг. Автоматична оцінка довіри до результатів тестування та біометрична верифікація особистості.
- мобільна версія. Можливість роботи на пристроях з різною роздільною здатнісю екрану та на мобільних Android та iOS.
- інтеграція SDK. Безшовна інтеграція із системою тестування, працює у браузері та не вимагає встановлення розширень, плагінів та стороннього ПЗ.
- можливість спілкування між учасниками та проктором (аудіо та відео чат).
- одночасне спостереження за значною кількістю студентів (до 30 учасників одночасно) на кожного проктора.

Даний застосунок може працювати в таких режимах:

Повна робота у хмарі з доступом через мережу Інтернет. Обслуговування, оновлення, технічна підтримка прихована, в свою чергу користувач отримує лише необхідний для роботи сервіс.

Розгортання на ПК тестованого. Забезпечення повного контролю за даними та процесами, та може працювати в локальній мережі без необхідності доступу до Інтернету [12].

Рисунок 1.6 – Приклад використання ProctorEdu

Exam Monitor – спеціалізований програмний за стосунок, який

використовується для спостереження за учнями та запобігання будь якого типу шахрайства під час проходження онлайн-тестування (рисунок 1.7).

Рисунок 1.7 – Головне вікно програми Exam Monitor

В процесі проведення спостереження за стосунок може реєструвати процеси, які в даний час виконуються на комп’ютері. Exam Monitor може відстежувати не лише активні процеси та відкриті файли але також і контент файлової системи [13].

До недоліків цієї програми можна віднести: обмежений функціонал та повна відсутність можливості контролю у режимі реального часу. Та необхідність постійного вивантаження інформації для подальшого ознайомлення з результатами спостережень.

Proctortrack – це програмне забезпечення для віддаленого стеження за підозрілою активністю студента під час тестування за допомогою веб-камери

Програмне забезпечення Proctortrack не має доступу до файлів на носіях інформації комп'ютера, проте має можливість отримувати доступ до різних типів файлів та процесів.

Після завершення тестування зібрані дані зберігаються на ПК користувачів, після чого відправляються на глобальні сервери для автоматичного аналізу [14].

Головним недоліком Proctortrack є те, що для роботи цього ПЗ необхідно придбати ліцензію.

1.5 Програмні засоби моніторингу

Такі програмні застосунки відносяться до інструментів цифрового прокторингу. Так наприклад данні інструменти можуть отримувати доступ до віддаленого комп’ютера користувача, створювати знімки екрану, забезпечувати одночасну роботу з декількома пристроями в режимі реального та ін.

Microsoft Remote Desktop - це клієнтський застосунок, який дає можливість користуватися ресурсами, файлами, та отримувати доступ до даних віддаленого комп’ютера або хост-компьютера. До основних можливостей віддаленого робочого столу можна віднести: можливість отримання повного доступу до робочого комп’ютера, використовувати програми встановлені на ПК, отримувати доступ до файлів та мережевих ресурсів. Основною перевагою даного застосунку є можливість використання практично з будь-якого пристрою, наприклад за допомогою смартфона або планшету.

Однак є і недоліки: обов’язкова наявність Windows Pro і вище на комп’ютері до якого підключаються (сервера) та неможливість підключення, в одній локальній мережі, пристроїв до віддаленого робочого столу без додаткових налаштувань [15,16].

Remote Utilities, представлена на ринку як Віддалений доступ RMS – являє собою одну з найпотужніших програм для віддаленого доступу одночасно до декількох комп’ютерів (рисунок 1.9). Включає в себе такі функції: можливість вибору режиму підключення (як через Інтернет, так і через RDP), встановлення програм та ПО дистанційно, отримання повного доступу до відеокамер, віддаленого реєстру і командному рядку, запис екрану віддаленого комп’ютера, функції чату (відео, аудіо, текстового), підтримка одночасної роботи з декількома моніторами [15,16].

UltraVNC – безкоштовний програмний застосунок для операційної системи Windows (рисунок 1.8). Для керування та роботи з віддаленим

комп’ютером використовується спеціальний протокол VNC. VNC (Virtual Network Computing) – мультиплатформний тип підключення з відкритим вихідним кодом, по своїй роботі схожий RDP.

Серед основних можливостей можна виділити: мультиплатформена передача файлів, можливість синхронізації буферу обміну, зчитування поєднань клавіш, текстовий чат для учасників робочої сесії.

Головним же недоліком даної програми є її складність в роботі та налаштуванні. Роботу з даним за стосунком складно назвати простою та інтуїтивно зрозумілою, особливо це стосується початківців [15,16].

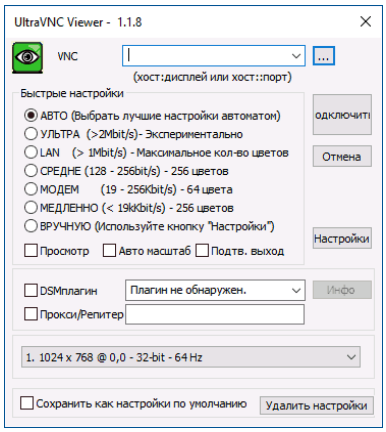

Рисунок 1.8 – Головне вікно програми UltraVNC

AeroAdmin – це спеціалізований програмний за стосунок призначений для віддаленого контрою до комп’ютерів через мережу Інтернет. В процесі роботи користувачу надається унікальний ідентифікаційний номер у системі, за яким адміністратор може знайти комп’ютер в мережі (рисунок 1.9).

Для підключення доступно декілька типів підключень: повне керування, лише перегляд та файловий менеджер.

Основною перевагою системи є її простота підключення, все що потрібно, це лише встановити програму після чого підключення буде увімкнено автоматично. Також незалежно, підключений ПК чи серверний, користувач може встановлювати обмеження та права доступу: тільки

перегляд, захоплення клавіатури та миші, файловий менеджер, синхронізація буфера обміну. В свою чергу адміністратор може додатково встановлювати права доступу для кожного учасника робочої сесії [15,16].

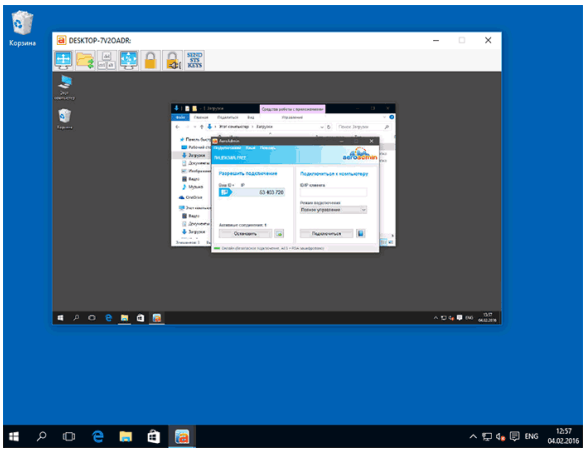

Рисунок 1.9 – Головне вікно програми AeroAdmin

DameWare – програмний застосунок створений як один з методів віддаленого контролю за комп’ютером користувача (рисунок 1.10). В першу чергу використовується як засіб технічної підтримки користувачів. Завдяки за стосунку спеціаліст технічної підтримки може оперативно отримати доступ до пристрою та усунути можливі проблеми [15,16].

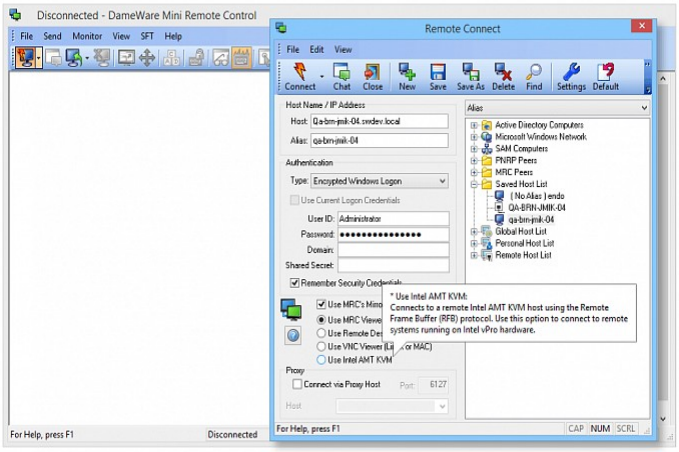

Рисунок 1.10 – Головне вікно програми DameWare

Ключовими особливостями даного за стосунку є:

– простота контролю. DameWare MRC забезпечує швидкий доступ до віддаленого ПК, без необхідності додаткових налаштувань.

– налаштування рівня доступу. Користувач може самостійно вікрити або навпаки заборонити доступ до свого ПК.

– можливість роботи при вимкненому ПК.

– наявність чату з оператором підтримки.

– відкритий доступ до створення знімків віддаленого екрану пристрою.

Серед недоліків можна виділити обмежений базовий фунціонал при використанні безкоштовної версії [15,16].

Спеціалізований програмний застосунок з широким асортиментом можливостей призначений для комплексного стеження за пристроями (комп’ютер, телефон, планшет і ті). Серед основних можливостей можна виділити:

– прослуховування та запис того що відбувається навколо цільового пристрою.

– можливість спостереження за користувачем за допомогою віддаленої камери.

– запис історії натискань клавіш для зазначених в налаштуваннях програмах.

– створення знімків в процесі роботи встановлених на пристрої застосунків.

– створення знімків екрану робочого столу.

– перегляд списку мережевих з'єднань.

– перегляд та запис відкритих закладок в усіх типах браузерів.

– Відстеження місцез находження.

Основним же недоліком є головна перевага застосунку, а саме багатогранне стеження за пристроєм. Через те що застосунок працює приховано та може бути встановлений без дозволу користувача, це може призвести до втручання у персональне життя людини [17].

Chrome Remote Desktop – являє собою ПЗ для віддаленого підключення до ПК користувачів за допомогою браузера Chrome або пристроїв Chromebook.

В процесі роботи проходить процес зчитування та передачі натискань клавіш клавіатури та миші, і пряма передача всії подій по мережі. Головним недоліком Chrome Remote Desktop являється те, що його робота неможлива без встановлення та використання веб-браузера Google Chrome [18].

AnyDesk – це програма для віддаленого доступу до ПК (рисунок 1.11). Головна перевага цієї програми – це висока швидкість роботи в порівнянні з усіма іншими аналогічними програмами.

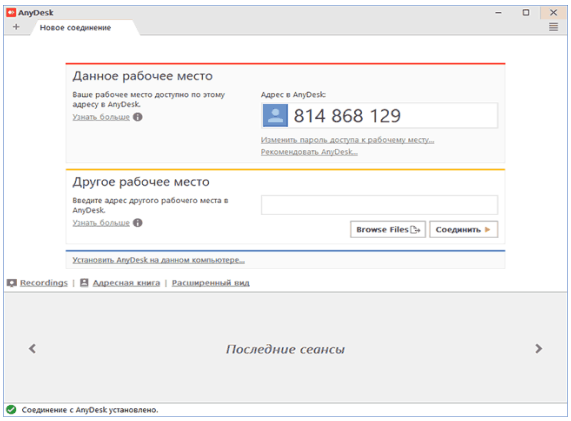

Рисунок 1.11 – Головне вікно програми AnyDesk

Завдяки цьому даний програмний продукт дає можливість віддалено адмініструвати комп'ютери, працювати в різних програмних засобах і виконувати безліч інших завдань, наприклад використовуватися в якості віддаленого спостерігача за тим, що відбувається на підконтрольних машинах, відстежувати будь-які зміни на екранах комп’ютерів, та швидко виявляти усі порушення [19].

До головних недоліків програми AnyDesk можна віднести: обмежений функціонал та збільшене навантаження на викладача через необхідність постійного спостереження за великою кількістю комп'ютерів.

1.6 Результати аналізу існуючих систем

Серед усіх розглянутих програмних рішень, які призначені чи можуть бути використані в якості програмного засобу контролю за процесом тестування, можна виділити такі недоліки:

– в процесі тестування студент знає про процес моніторингу, система не має можливості підтримки процесу контролю в режимі реального часу

– обмежений доступний для використання функціонал при використані безкоштовної ліцензії на продукту.

– відсутність для студентів можливості ознайомлення із зібраними в процесі спостереження результатами після закінчення самого тестування.

– платна ліцензія.

Результати аналізу існуючих рішень представлені в таблиці 1.1. Розглянутим аналогам відповідають:

– Екзамус.

– Exam Cookeys.

– Exam Monitor.

– Proctortrack.

– AnyDesk.

– Chrome Remote Desktop.

– Spyera.

– DameWare.

– AeroAdmin.

– UltraVNC.

– Віддалений доступ RMS, Remote Utilities.

– Microsoft Remote Desktop.

– ProctorEdu.

Знак «+» – для відображення наявності конкретного функціоналу означає наявність функціоналу, знак «-» – відсутність або неповну підтримку даної можливості.

Таблиця 1.1 – Порівняння існуючих рішень

Характеристики	Програмні рішення												
	1	2	3	4	5	6	7	8	9	10	11	12	13
Відстеження буферу обміну	–	–	–	–	–	–	+	–	–	–	–	–	–
Відстеження активних процесів	–	+	–	–	–	–	+	+	–	–	–	–	–
Створення знімків робочої області екрану комп'ютера	–	–	+	–	–	–	+	–	+	–	–	–	–
Створення знімків з веб камери	+	–	–	–	–	–	+	–	–	–	–	–	–
Отримання активного URL в різних типах браузерів	–	–	–	–	–	–	+	+	–	–	–	–	–
Отримання переліку натиснутих клавіш на клавіатурі	–	–	–	–	–	–	+	–	+	–	–	–	–
Отримання переліку мережевих підключень та процесів	–	–	+	–	–	–	+	–	–	–	–	–	–
Можливість використання веб камери для постійного спостереження	+	–	–	+	+	+	+	+	–	–	–	–	+
Відстеження процесу тестування в режимі реального часу	+	–	–	+	+	+	+	–	+	–	+	+	+
Можливість відправки повідомлення конкретному студенту	–	–	–	–	–	–	+	–	–	–	–	–	–
Безкоштовна ліцензія	–	–	–	–	–	–	+	–	–	–	+	+	–
Можливість роботи на мобільних пристроїв	–	+	–	–	–	–	+	–	–	–	–	–	+
Інтеграція із системою тестування	–	–	+	–	–	–	+	+	+	+	–	–	–
Можливість спілкування між учасниками та проктором	+	–	–	–	–	–	+	–	–	–	–	–	–

Продовження таблиці 1.1

Можливість після закінчення тесту ознайомитись зі своїми результатами	–	–	–	–	–	–	+	–	–	–	–	–	–
Налаштування рівня доступу користувачем.	–	–	+	–	–	–	+	–	–	–	+	+	–
Можливість роботи при вимкненому ПК;	+	–	–	+	+	+	+	+	+	+	–	–	–
Відстеження місце знаходження	–	–	–	–	–	–	+	–	–	–	–	–	
Прослуховування та запис того що відбувається навколо пристрою	–	–	–	–	–	–	+	–	–	–	–	–	
Стеження за положенням обличчя в процесі проходження е-тестування	+	–	–	–	–	–	+	+	+	+	–	–	–
Відстеження поворотів голови та детектування втрати з кадру	–	–	–	–	–	–	+	–	–	–	–	–	–

1.7 Постановка задачі

Метою кваліфікаційної роботи є створення системи спостереження за поведінкою студентів, під час проведення е-тестів.

Для досягнення поставленої мети необхідно виконати наступні завдання:

– дослідити існуючі спеціалізованих системи контролю, стеження та повного контролю за процесом проходження е-тестування, в режимі реального часу.

– дослідити основні переваги та недоліки існуючих систем та визначити найбільш вдалі функціональні рішення.

– розробити модель взаємодії клієнтами та сервером в процесі тестування.

– проаналізувати результати, отримані під час тестування.

2 СИСТЕМА АНАЛІЗУ ПОВЕДІНКИ КОРИСТУВАЧА ПІД ЧАС ПРОВЕДЕННЯ Е-ТЕСТІВ

2.1 Огляд засобів розробки

Стек технологій – це набір інструментів, що застосовується при розробці проектів і включає мови програмування, фреймворки, компілятори і т.д. Від обраного розробником стеку технологій напряму залежить продуктивність роботи застосунку, вимоги до апаратних ресурсів, надійність роботи, захищеність програмного забезпечення і т. ін. [20].

Серед технологій для створення десктопних застосунків на даний час найбільшого поширення набули такі засоби, як C/C++, C#, Python, Java. Для розробки віконних застосунків за допомогою мови C# використовується технологія WinForms. Реалізація можливостей серверної частини програми здійснюється за допомогою стандартних засобів PHP, JS, HTML, CSS і т.і.

Інтерфейс користувача та основні можливості взаємодії з ним, реалізується за допомогою CMS Wordpress. Ця технологія дозволяє швидко та ефективно створити зручний і багатофункціональний інтерфейс з можливістю швидкого налаштування внутрішнього контенту. Можливість обміну даними між сервером та клієнтом реалізована завдяки використанню серверної платформи Node.js. Використання подієво-орієнтованої моделі та не блокуючого процесу введення/виведення в Node.js дозволяє досягти більш високої ефективності роботи та простоти використання.

Для зберігання зібраних в процесі роботи даних, незалежно від типу програми широко використовуються різні типи реляційних систем керування базами даних (СКБД), до найпопулярніших можна віднести SQLite, MySQL та PostgreSQL.

В даному проекті для зберігання даних на клієнтських машинах використовується портативна СКБД SQLite – завдяки простоті і швидкості

роботи вона найкраще підходить як локальне сховище клієнтських даних [21,22]. Для серверної частини застосунку використовується СКБД MySQL [23]. Завдяки схожій структурі типів даних в цих СУБД, можна легко організувати синхронізацію та обмін даними між ними без необхідності використання сторонніх засобів синхронізації [24]. Однак при цьому необхідно забезпечити правильний підхід в синхронізації та їх взаємодії в клієнт-серверному застосунку.

2.2 Технології реалізації клієнтської частини застосунку

2.2.1 C# та платформа WinForms

C# – проста, сучасна, об'єктно-орієнтована мова програмування, що забезпечує безпеку типів. Мова C# спроектована та розроблена спеціально для застосування з Microsoft .NET Framework (багатофункціональної платформи розробки, розгортання та виконання розподілених застосунків). До основних властивостей C#, завдяки яким відкривається можливість створення надійних та стійких програм, можна віднести: обробка виняткових ситуацій, що забезпечує структурований і розширюваний підхід до виявлення помилок під час роботи програми, можливість автоматичного прибирання сміття, яка автоматично звільняє пам'ять, зайняту не використовуваними об'єктами; структура мови, що забезпечує безпеку типів, унеможливлює отримання значення неініціалізованих змінних, індексувати масиви поза їх межами або виконувати безконтрольне приведення типів. C# забезпечує мовні конструкції, які безпосередньо підтримують ці концепції, що робить її дуже природною мовою для створення і застосування компонентів програмного забезпечення [25,26].

Для створення графічних інтерфейсів за допомогою платформи .NET застосовують різні технології, але в основному – Window Forms та WPF (для ОС Windows 8 / 8.1 / 10). Завдяки простоті вивчення та застосування Window

Forms досі залишається найбільш популярною платформою [26]. Використання Windows Forms дозволяє створювати програмні застосунки з повнофункціональним графічним інтерфейсом, та можливістю працювати за умов відсутності підключення до Internet [27].

2.2.2 Середовище розробки Microsoft Visual Studio

Microsoft Visual Studio – інтегроване середовище розробки, що призначене для написання, налагодження та виконання коду, а також подальшої публікації застосунків. Крім стандартного редактора та відладчика, які існують в більшості IDE, Visual Studio включає в себе компілятори, засоби авто завершення коду, графічні конструктори і багато інших функцій для спрощення процесу розробки різноманітних застосунків [28].

Visual Studio включає в себе такі елементи:

– редактор вихідного коду з можливістю найпростішого рефакторінгу і підтримкою технології IntelliSense;

– вбудований відладчик, що має можливість роботи як відладчик рівня вихідного коду, так і при необхідності як відладчик машинного рівня;

– простий редактор форм, який спрощує створення графічного інтерфейсу;

– вбудований веб-редактор, дизайнер класів і дизайнер схеми бази даних [28].

Для розширення функціональності Visual Studio надає можливість створювати та підключати сторонні застосунки, включаючи підтримку систем контролю версій вихідного коду (як, наприклад, Git), додавання нових наборів інструментів (наприклад, для редагування і візуального проектування коду на предметно-орієнтованих мовах програмування), або інструменти для інших аспектів процесу розробки програмного забезпечення (наприклад, використання клієнту Team Explorer) [29].

2.2.3 Розпізнавання облич за допомогою методу Віоли Джонсона та Каскаду Хаара

Каскад Хаара – являє собою спеціальний набір масок та прямокутних зображень (фреймів), кожен з яких представляє є зображенням в якому попередньо було створено чорно-білий узор (комбінація чорних і білих частин в різних положеннях). В навчальному наборі каскаду одночасно може бути необмежена кількість таких фреймів (рисунок 2.1). Складність узорів визначається в залежності від задачі розпізнавання і може відрізнятися [30].

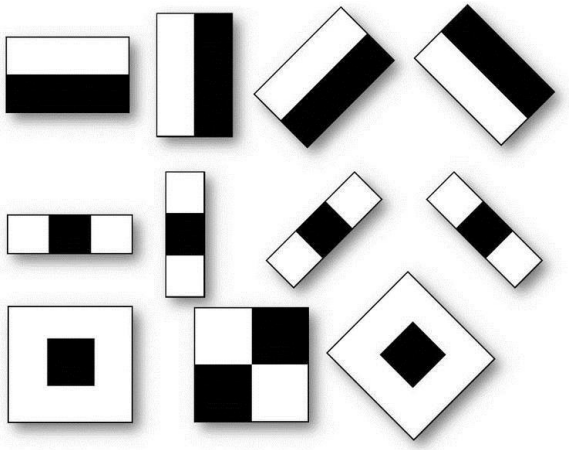

Рисунок 2.1 – Примітиви Хаара

Створені маски (фрейми) накладються на різні частини кадру, завдяки чому і проводиться процес детектування об’єкту в кадрі. Після накладання маски на певну частину отриманого кадру в систему повертається числове значення, а саме результат згортки маски з кадром: програма складає яскравість всіх пікселів зчитаного зображення, що потрапили в білу частину маски при накладенні, а також яскравість тих пікселів, що потрапили до чорної частини маски, після чого обчислює різницю цих значень порівнюючи результат з граничною величиною.

Серед основних принципів, на яких заснованих метод Віоли Джонсона

можна виділити:

– використовуються зображення в інтегральному поданні, що дає змогу обчислювати швидко необхідні об'єкти;

– використовуються ознаки Хаара, за допомогою яких відбувається пошук потрібного об'єкта (в даному контексті, особи та її рис);

– використовується бустинг (від англ. boost - поліпшення, посилення) для вибору найбільш підходящих ознак для об'єкта, що шукається на даній частині зображення;

– всі ознаки надходять на вхід класифікатора, що дає результат «правильно» чи «брехня»;

– використовуються каскади ознак для швидкого відкидання вікон, де обличчя не знайдено [30, 31].

На початку роботи для навчання класифікаторів необхідно витратити велику кількість часу. Але при цьому отримання результатів розпізнавання особі проводиться дуже швидко. Саме тому і був обраний даний метод розпізнавання обличчя в відео потоці. Використання методу Віоли-Джонсона і ознак Хаара є одним із найкращих варіантів за співвідношенням показників ефективність розпізнавання/швидкість роботи [32].

Також даний метод має таку особливість, а саме вкрай низьку ймовірність помилкового виявлення обличчя. Алгоритм детектування дуже добре працює з рисами обличчя під невеликим кутом, приблизно до 30 градусів. При зміні кута нахилу більше ніж 30 градусів, відсоток виявлень і точність розпізнавання різко падає. Дана особливість дозволяє в стандартній реалізації детектувати втрату з поля зору поверненого обличчя тестованого, що значною мірою спрощує використання алгоритму в системах контролю за студентами [33,34]. Для перевірки було проведено приблизно 400 експериментів, в ході яких були спеціально створені такі умови, які б найбільш точно показали працеспроможність алгоритму пошуку облич. А саме в такі часові рамки: ранок, день, вечір та при штучному освітленні (Штуч осв) Та були отримані такі усереднені результати (Таблиця 2.1).

Таблиця 2.1 – Результати експериментів

№	Час доби	Дистанція обличчя до камери	Кількість людей у сцені	Позиція обличчя відносно камери	Час пошуку	Точність
1	Ранок	20 см	1	0° - 45°	0.2 секунди	95%
2	Ранок	50 см	2	0° - 45°	0.6 секунди	95%
3	Ранок	100 см	3	0° - 45°	1.4 секунди	89%
4	Ранок	20 см	1	45° - 90°	2.2 секунди	74%
5	Ранок	50 см	2	45° - 90°	2.6 секунди	60%
6	Ранок	100 см	3	45° - 90°	4.9 секунди	43%
7	День	20 см	1	0° - 45°	0.1 секунди	95%
8	День	50 см	2	0° - 45°	0.2 секунди	95%
9	День	100 см	3	0° - 45°	0.9 секунди	90%
10	День	20 см	1	45° - 90°	1.8 секунди	91%
11	День	50 см	2	45° - 90°	2.1 секунди	85%
12	День	100 см	3	45° - 90°	3.2 секунди	79%
13	Вечір	20 см	1	0° - 45°	1.1 секунди	87%
14	Вечір	50 см	2	0° - 45°	1.9 секунди	73%
15	Вечір	100 см	3	0° - 45°	3.1 секунди	79%
16	Вечір	20 см	1	45° - 90°	2.6 секунди	71%
17	Вечір	50 см	2	45° - 90°	4.7 секунди	63%
18	Вечір	100 см	3	45° - 90°	7.9 секунди	57%
19	Штуч осв	20 см	1	0° - 45°	0.3 секунди	95%
20	Штуч осв	50 см	2	0° - 45°	1.6 секунди	73%
21	Штуч осв	100 см	3	0° - 45°	8.6 секунди	0%
22	Штуч осв	20 см	1	45° - 90°	Не знайдено	0%
23	Штуч осв	50 см	2	45° - 90°	Не знайдено	0%
24	Штуч осв	100 см	3	45° - 90°	Не знайдено	0%

2.2.4 СКБД SQLite

SQLite – компактна вбудована реляційна СКБД, яка є однією з

найбільш використовуваних систем управління базами даних. Висока швидкість можлива завдяки внутрішній архітектурі та відсутності необхідності в з'єднаннях типу «сервер-клієнт» і «клієнт-сервер».

Для СКБД SQLite можна виділити такі особливості, як висока надійність та простота використання, через те, що база даних являє собою звичайний локальний файл, який можна переміщати разом з усіма файлами програми. Але головною перевагою SQLite перед іншими СКБД є те, що для бази даних не потрібно сервера. Це означає, що ядро SQLite не є окремим процесом, з яким працює програма, а являє собою бібліотеку яка стає частиною самої програми під час її виконання. Для роботи з СКБД в Visual Studio необхідно до проекту підключити бібліотеку System.Data.SQLite. Дана бібліотека включає в себе не тільки класи .NET Framework, але і весь код самої СКБД SQLite, що спрощує процес її використання, та надає можливість прискорити написання програми [35, 36].

2.2.5 Бібліотека AForge.NET

AForge.NET – це бібліотека з відкритим вихідним кодом, розроблена на мові C#. Ця бібліотека використовується при вирішенні завдань, пов'язаних з комп'ютерним зором та в галузі штучного інтелекту. Діапазон засобів, що застосовуються бібліотекою, досить різноманітний: обробка зображень, нейронні мережі, генетичні алгоритми, нечітка логіка, машинне навчання, робототехніка та багато іншого. Основні компоненти бібліотеки AForge.NET:

– AForge.Imaging – бібліотека, призначена для роботи з зображеннями і фільтрами;

– AForge.Vision – бібліотека, яка застосовує методи комп'ютерного зору;

– AForge.Video – пакет бібліотек для виконання робіт, пов'язаних з відеоданими;

– AForge.Neuro – бібліотека, в якій використовуються можливості

нейронних мереж;

– AForge.Genetic – бібліотека, що призначена для вирішення різноманітних завдань із застосуванням генетичних алгоритмів;

– AForge.Fuzzy – бібліотека, яка працює з нечіткою логікою;

– AForge.Robotics – бібліотека, що підтримує методи, використовувані в області робототехніки;

– AForge.MachineLearning – бібліотека, в якій застосовуються елементи машинного навчання.

В ПЗ бібліотека використовується для захоплення відео з веб-камери, для цього використовується клас AForge.Video та AForge.Video.DirectShow. Для цих класів необхідно вказати маркер пристрою, з якого буде здійснюватися захоплення, та призначити обробник події NewFrame [37].

2.2.6 System.Windows.Automation

Microsoft UI Automation – це інтерфейс прикладного програмування (API), використання якого дозволяє отримувати доступ, ідентифікувати і маніпулювати елементами призначеного для користувача інтерфейсу (UI) іншої програми (рисунок 2.2).

UI Automation забезпечує програмний доступ до більшості елементів інтерфейсу застосунків на робочому столі користувача, дозволяючи іншим продуктам, таким як програми читання з екрану, надавати кінцевому користувачеві інформацію про інтерфейс користувача і керувати ним за допомогою засобів, відмінних від стандартного введення UI [38].

Специфікація автоматизації користувальницького інтерфейсу, забезпечує більш гнучкий програмний доступ до елементів користувальницького інтерфейсу на ПК користувача, дозволяючи іншим технологічним застосункам, таким як програми зчитування інформації з екрану, передавати дані про інтерфейс користувача і керувати користувальницьким інтерфейсом віддалено. Дана специфікація не

підтримується на платформах, відмінних від Microsoft Windows.

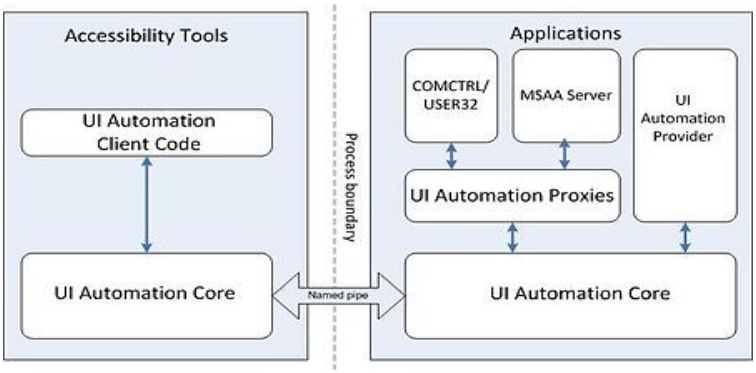

Рисунок 2.2 – Схема роботи UI automation

Реалізація цієї специфікації Windows також називається UI Automation. Автоматизація інтерфейсу користувача ширше, ніж просто визначення інтерфейсу.

Об'єктна модель реалізації та функції, які дозволяють клієнтам легко отримувати інформації про події, отримувати значення властивостей та маніпулювати елементами інтерфейсу ПК [39].

Готова інфраструктура для пошуку та маніпулювання даними, через мережу процесів. Підготовлений набір інтерфейсів для наглядного відображення структури та властивостей програм.

Набір підготовлених інтерфейсів для користувачів та провайдерів для стеження за властивостями та функціями специфічних інтерфейсів.

Якщо розглядати архітектуру UI Automation, можна виділите те, що в процесі роботи проводиться пряме завантаження компонентів UI Automation Core до процесів клієнтських застосунків (рисунок 2.2). Управління компонентами та програми надає можливість спеціалізованої міжпроцессорної взаємодії між UI та кінцевим користувачем. Даний компонент являє собою один із основних інструментів, що забезпечує вибірку та кешування властивостей, і покращує міжпроцесорну продуктивність [38,39].

2.2.7 Emgu.CV

Emgu CV — це бібліотека яка являє собою кросплатформенну «обгортку» для .NET обробки зображень OpenCV (основної бібліотеки), Emgu CV також називають бібліотекою машинного зору [40] (рисунок 2.3).

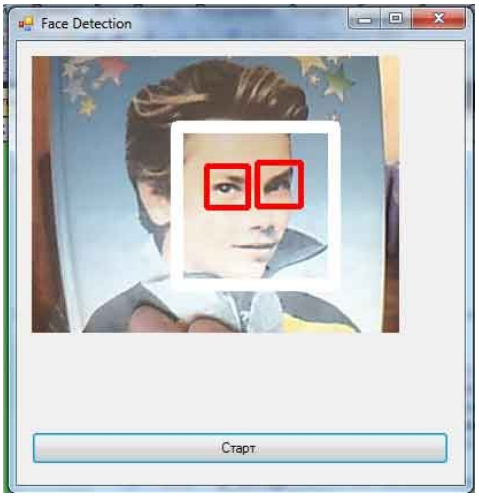

Рисунок 2.3 – Приклад використання бібліотеки Emgu.CV для детектування обличчя на фотографії

Основною задачею даної бібліотеки є вирішення різноманітних завдань пов’язаних з 2D та 3D графікою, а саме розпізнаванням осіб та предметів по фото, відео та ін. На даний момент існує декілька основних напрямів, в яких використовується дана бібліотека [41].

Робота з 2D-зображеннями.

В цьому випадку функції алгоритму працюють з особливими точками об’єкта та відстанями між ними. Ці точки та відстані унікальні, тому, порівнюючи їх з відповідними точками та відстанями еталонів у базі даних, можна зробити висновок, чи відповідає об’єкт у кадрі еталонному об’єкту на фотографії. Обчислюється міра схожості у фреймі, після чого дані більш схожого зображення видаються на екран [40].

Використання такого методу дає можливість ідентифікувати

зображення у високій роздільній здатності та без засвічення. Чутливість алгоритму до поворотів та нахилів, освітлення висока, тому він підходить для розпізнавання людей у неорганізованих потоках (у натовпі, на вулиці тощо).

Побудова 3D-моделей

Існують спеціалізовані методи, що вирішують проблему за допомогою побудови 3D-моделей об'єкта у просторі. Метод потребує встановлення кількох спеціальних стерео камер, синхронізованих між собою. З появою в зоні спостереження детектованого об’єкту, камери роблять серію знімків з різних ракурсів. Потім будується 3D-модель задетектованого об’єкту, і проводиться робота з особливими точками та відстанями між ними [41].

2.3 Технології реалізації серверної частини застосунку

2.3.1 CMS WordPress

CMS WordPress – це одна з найпопулярніших систем управління контентом (Content Management System), що застосовується в світовій практиці швидкої розробки веб-застосунків. Платформа написана на скриптовій мові програмування PHP, що використовується для розробки різних web-сервісів, являється повністю безкоштовною, та має повністю відкритий вихідний код.

Вбудована система «тем» і «плагінів», разом із вдалою архітектурою дозволяє за допомогою WordPress створювати будь-які типи сайтів, від простих блогів до досить складних повноцінних ресурсів (наприклад, Інтернет магазинів) [42].

CMS WordPress характеризується наступними особливостями:

– простота використання;

– багатий функціонал;

– відсутність необхідності будь-яких особливих навичок для обслуговування;

– можливість створювати публікації з використанням сторонніх програм;

– наявність і підтримка великої кількості бібліотек, готових плагінів і модулів;

– можливість керування вмістом сайту з будь-якої точки світу;

– доступність, безкоштовність (ліцензія GNU / GPL);

– відсутність спеціальних вимог до серверу;

– швидкість та простота установки;

– наявність великої кількості безкоштовних стилів і «дизайнів»;

– постійне оновлення та підтримка користувача;

– постійне вдосконалення системи захисту.

Досвідчені користувачі можуть розширити базові можливості шаблону за рахунок використання додаткових плагінів, які у величезній кількості можна знайти на просторах Мережі або у вмонтованому магазині плагінів WordPress [42, 43, 44].

2.3.3 СКБД MySQL

MySQL – СКБД, в якій всі дані зберігаються в окремих таблицях, завдяки чому досягається виграш в швидкості та гнучкості запитів. Таблиці зв'язуються між собою за допомогою відносин, завдяки чому забезпечується можливість об'єднувати при виконанні запиту дані з декількох таблиць. В порівнянні з іншим засобами управління базами даних, MySQL є дуже швидкою, надійною та простою у використанні [47]. Для роботи з базами даних MySQL у програмах на C# необхідно підключити бібліотеку MySql.Data, в якій містяться всі необхідні функції для взаємодії між застосунком на C# і сервером MySQL [48]. Для роботи з віддаленими серверами баз даних MySQL необхідно додатково встановити на комп'ютер Connector/NET 8 або вище. Connector/NET – це програмний засіб, який написаний на мові програмування C#, та призначений для надання

застосункам на платформі .NET отримувати доступ до високопродуктивного та безпечного підключення до віддалених даних, без необхідності створення окремих локальних серверів для розміщення баз даних з MySQL [49,50].

2.3.4 Плагін Carbon Fields

Carbon Fields – це плагін, який дозволяє легко створювати призначені для користувача (мета) поля в панелі адміністрування WordPress. Це дозволяє розробникам теми пов'язувати мета інформацію з різними об'єктами на сайті WordPress (такими як повідомлення, терміни таксономії, віджети і т.д.). Основним завданням Carbon Fields є створення більш зручною для розробників WordPress платформи веб-розробки [43].

Цей плагін є альтернативою ACF. ACF – плагін який надає можливість створити розширений інтерфейс користувача та дає можливість створення користувацьких полів в WordPress [42]. Але незважаючи на багатий функціонал, простоту використання, та наявність можливості розширеного налаштування, повний функціонал ACF являється платним, в свою чергу Carbon Fields дає ще більш широкий асортимент можливостей абсолютно безкоштовно. Використання цього плагіну дає можливість створювати складні ресурси з можливістю адаптивної роботи з їх вмістом, а саме: адаптації всього сайту під будь-яку необхідну мову, можливість вільного додавання нових полів в уже існуючі блоки, та дуже простий спосіб їх налаштування. Для роботи з Carbon fields його можна встановити через composer (вимагає htmlburger / carbon-fields) або пакетно. Плагін підтримує PHP 5.3 або вище і використовує для роботи стандартний простір імен PHP, що спрощує та прискорює роботу при написанні застосунку [43].

Основними компонентами бібліотеки є:

- Container – представляє і управляє групою полів.
- Field – являє собою одне поле.
- Data Storage – керує базовим сховищем даних для значень полів.

2.4 Протокол WebSocket

Протокол WebSockets – це передова технологія, яка дозволяє створювати інтерактивне з'єднання між клієнтом (браузером) та сервером для обміну повідомленнями в режимі реального часу (рисунок 2.4) .

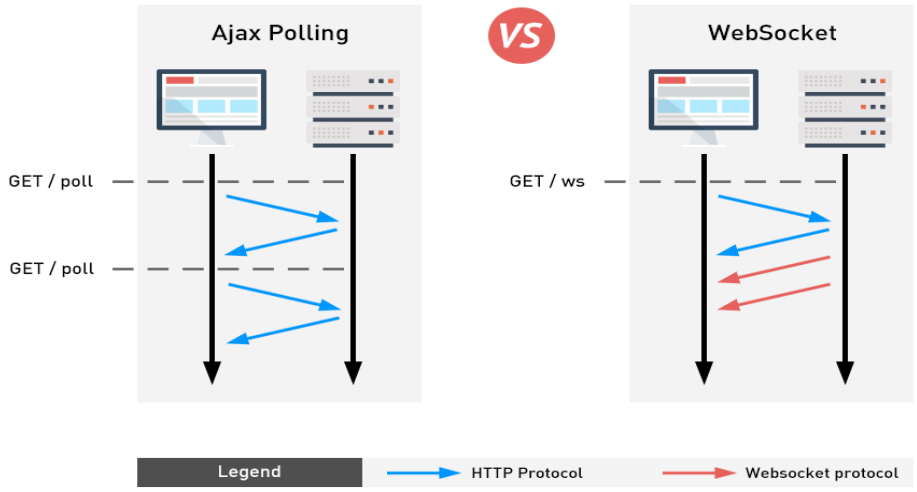

Рисунок 2.4 – Схема обміну повідомленнями між клієнтом та сервером при використанні протоколу HTTP та WebSoket

Для реалізації захищеного обміну повідомленнями між клієнтом і сервером, найчастіше використовують протоколи WebSocket Secure і HTTPS. Основною відмінністю цих протоколів, є використання різних абсолютно принципів обміну та передачі даних, так як на відміну від HTTP, протокол WebSocket є двонаправленим. Але при цьому більшість проблем зв’язаних з безпекою, які виникають в веб- застосунків, реалізованих на основі Websockets, такі як атака MITM (Man in the Middle) (рисунок. 2.5), реалізація аутентифікації і авторизації, також відносяться і до застосунків що використовують протокол HTTP [51]. Це може дуже сильно вплинути на поведінку як веб-проксі так і брандмауерів, оскільки більшість з них в перевіряють пакети, визначаючи їх заголовки [51,52].

Основна ж захисна стратегія включає в себе: Перевірку довжини корисного навантаження, це необхідно щоб уникнути переповнення буфера. Уникнення виснаження ресурсів. Наприклад, виділення пам'яті без перевірки розміру введених даних в буфер. Запобігання відправки повідомлень клієнтом в неправильному порядку. Закривати з’єднання у разі отримання будь-яких непередбачуваних даних.

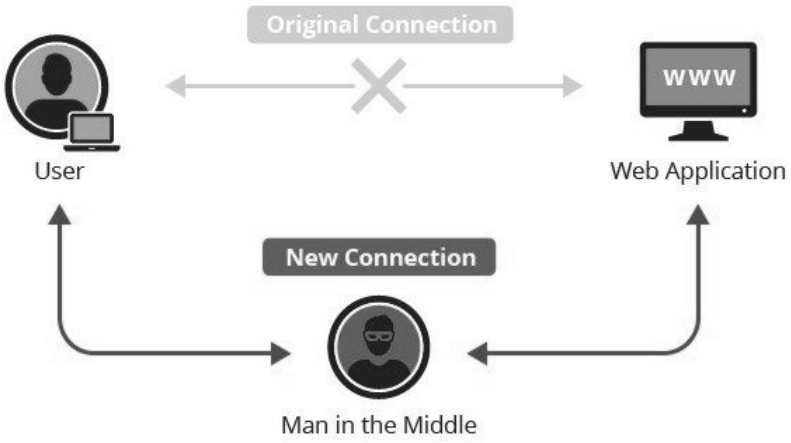

Рисунок 2.5 – Атака MITM(Man in the Middle)

Даний підхід також застосовується і до клієнтів [52]. А оскільки з'єднання з клієнтом і сервером не закривається (тримається відкритим постійно), це дозволяє уникнути передачі зайвих даних, як при використанні HTTP-заголовків. До основних переваг протоколу WebSockets можна віднести:

– стійке та стабільне двостороння з'єднання;

– зниження використовуваного трафіку;

– забезпечення безпечної передачі даних;

– мінімальна затримка в процесі передачі інформації від клієнта до сервера, та навпаки.

Так само в стандарті WebSockets немає обмежень за кількістю відкритих з'єднань і по черговості запитів [51, 52].

2.5 Модель системи спостереження за поведінкою студентів під час проведення е-тестів

Процес спостереження за поведінкою студентів під час проведення е-тестів може бути поданий 3-х рівнева структура, що описується моделлю (2.1), яка включає такі компоненти: клієнтська частина, блок синхронізації з сервером та серверна частина (рисунок 2.6).

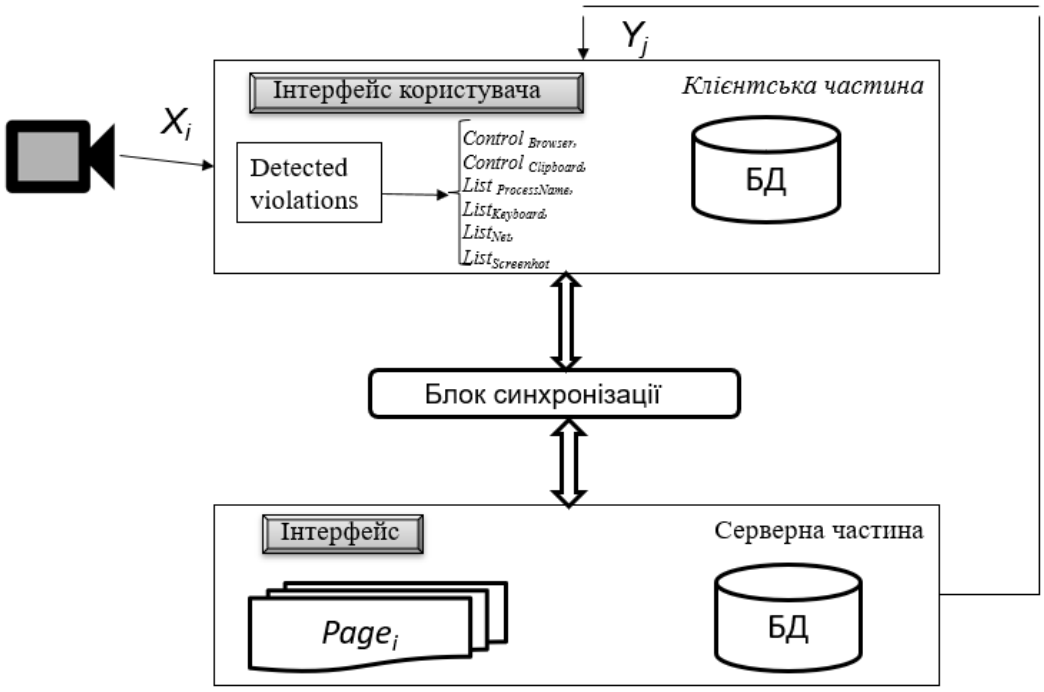

Рисунок 2.6 – Модель системи спостереження за поведінкою студентів під час проведення е-тестів

Модель процесу спостереження за поведінкою студентів під час проведення е-тестів виражається як перетворення вхідних значень X у вихідні величини Y:

$$Z \subset X \times Y, \tag{2.1}$$

Y, $(Y \subset \Upsilon)$ – множина формалізованих властивостей. Таким чином,

простір (універсум) $\Sigma = X \times Y$ включає $Z \subset H \times Y$, це означає, що існує така підмножина H, ($H \subset X$) та відношення між ними, на яких будується модель Z, $(Z \subset \Sigma)$.

Для вихідної величини Y побудована множина завдань, розв'язання яких належить множині D_Z={D_{user}, D_{sin}, D_{server}}: тут D_{user} ={$Control_{Browser}$, $Control_{Clipboard}$, $List_{ProcessName}$, $List_{Keyboard}$, $List_{Net}$, $List_{Screenhot}$}, де $Control_{Browser}$ – завдання визначення активного URL в різних типах браузерів, $Control_{Clipboard}$ – завдання відстеження змін вмісту буфера обміну, $List_{ProcessName}$ – завдання отримання списку активних процесів на комп'ютері користувача, $List_{Keyboard}$ – завдання обробки та логування натиснутих користувачем клавіш на клавіатурі, $List_{Net}$ – завдання отримання списку мережевих процесі, $List_{Screenhot}$ – завдання створення скріншотів всієї області екрану комп'ютера та знімків за допомогою веб-камери, при зміні активного вікна програми або його розмірів; D_{sin} -завдання синхронізації клієнтської та сервеної частини; D_{server} ={ $Page_i$}, i=1,2,3, де $Page_1$ – завдання створення сторінки, де відображається інформація про всі дії, які відбуваються на клієнтських машина, $Page_2$ – завдання створення сторінки роботи з текстовими даними дозволяє ознайомитися з текстовими результати, які були зібрані за час спостереження, $Page_3$ – завдання створення сторінки роботи з зображеннями дозволяє користувачеві ознайомитися з фотознімками та скріншотами, які були зроблені під час тестування.

Відображення $T: X_D \to Y_D$ дозволяє для кожного $X_D(i)$ знайти таке $Y_j \in Y_D$ ($j = \overline{1,Q}$, Q – кількість тестованих), що є розв'язком завдання D_X.

Значення $Y_j \in Y_D$ використовуються для прийняття рішення про порушення.

Користувацький інтерфейс являє собою програму, де на вхід подається зчитаний з серверу набір інструкцій, за допомогою яких активуються необхідні для спостереження функції. Зчитана інформація передається в процедурний блок. Інформація в ньому являє собою матрицю функцій детектування та пошуку порушень. В процесі обробки інформації, функція детектування зрівнює зчитані результати з попередніми, та еталонними (передаються одночасно з командами з серверу). На виході результат представляє собою згрупований масив, який містить в собі таку інформацію: ідентифікатор користувача, повідомлення про порушення таблицю БД на комп'ютері студента, в яку необхідно записати задетековане порушення, таблицю БД на сервері, та інформацію про порушення. Процес прийняття рішення про порушення представляє собою «порівняння» отриманих проміжних результатів від функцій застосунку з попередніми, та додатково з еталонними значеннями, переданими за серверу перед початком тестування. На сервері після отримання повідомлення, можна ознайомитися з самим порушенням, для цього реалізований спеціальний функціональний web-інтерфейс. Та в разі необхідності проктор може відправити повідомлення з попередженням конкретному студенту, або при необхідності змінити налаштування клієнтських застосунків.

Для оцінки якості процесу функціонування пропонованої системи використовується сукупність критеріїв оцінки ефективності $K = \{k_1, k_2\}$, що дозволяють визначити реакцію дететкування порушень.

Першим критерієм є час дететкування порушень k_1:τ=min(τ^{Br}, τ^{Cl}, τ^{Pr}, τ^{key}, τ^{Net}, τ^{Scr}),

де τ^{Br} – час визначення активного URL в різних типах браузерів,

τ^{Cl} – час відстеження змін вмісту буфера обміну,

τ^{Pr} – час отримання списку активних процесів на комп'ютері користувача,

τ^{key} – час обробки та логування натиснутих користувачем клавіш на клавіатурі,

τ^{Net} – час отримання списку мережевих процесі,

τ^{Scr} – час створення скріншотів всієї області екрану комп'ютера та знімків за допомогою веб-камери, при зміні активного вікна програми або його розмірів.

Співвідношення множини реально створених сторінок $Page_i$ $(i=1,2,3)$ характеризуються такими характеристиками: $\psi_i=\{\rho_i, e_i\}$,

де $\rho_i = \frac{a}{a+c}$ – коефіцієнт повноти, який характеризує інформацію про дії на клієнтських машинах, що відображена на побудованих сторінках $Page_i$ до загальної кількості дій на клієнтських машинах; $e_i = \frac{b}{a+b}$ – коефіцієнт шуму, що характеризує частку наданої інформації без порушень; a – кількість спостерігальних порушень; b – кількість поведінки без порушень; c- кількість загальних порушень.

Для ефективного функціонування пропонованої моделі необхідно, щоб система відповідала таким вимогам:

$$k_2 : \forall(D_j \in D_Z)[(\tau < \tau^{max})\&(\rho_i \to 1)\&(e_i \to 0)] \Rightarrow Y_{D,}$$

де D– підзавдання загального завдання D_Z; τ^{max} – максимально допустимий час вирішення завдання.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Системні вимоги

Серверна та клієнтська частина програми для моніторингу тестування студентів призначена для використання на ПК під управлінням ОС Windows 10 (32/64-розрядні версії). В ОС повинен бути встановлений Connector/NET 8.0.20 або вище, та NET Framework версії 4.5 або вище.

Розроблюваний програмний за стосунок має можливість працювати в повністю автономному режимі, за винятком ситуацій, коли необхідно виконати зміну порту підключення до серверного за стосунку. Також ПЗ має можливість віддаленого контролю за всіма підключеними користувачами та дозволяє відправляти конкретним користувачам повідомлення про порушення.

3.2 Опис функцій розробленого програмного забезпечення

Основна задача програми – можливість спостереження за процесом тестування в режимі реального часу так і в режимі перегляду результатів спостережень. Програма являє собою клієнт-серверний застосунок, в якому клієнтська частина повинна працювати у фоновому режимі та підтримувати такий функціонал:

Відстеження активних URL в найпоширеніших типах браузерів такі параметри як: розпізнавання активного типу браузера, його назва та вміст рядка адреси.

Стеження та відстеження в режимі реального часу буд-яких змін параметрів (ширини та висоти) активного вікна. Для виявлення порушень виконується порівняння еталонних параметрів (отримуються одразу після запуску) через заданий в налаштуваннях інтервал. В разі виявлення змін,

отримані нові параметри ширини та висоти вікна записуються та відправляється повідомлення на сервер.

Зчитування списку активних процесів на комп’ютерів тестованого може виконуватися як з певним інтервалом так і по команді проктора. Можливість виявляти та ознайомлюватися зі списком активних в процесі тестування процесів, відкриває можливість виявляти будь-які сторонні програми та утиліти на робочому ПК студента.

Отримання переліку мережевих підключень – дана функція дозволяє захиститися від можливого проходження тесту за допомогою систем віддаленого управління.

Автоматичне зчитування місту буфера обміну та ведення логу натиснутих клавіш (за винятком клавіш миші). Дана функція дозволяє відстежити та оперативно запобігти будь-яким спробам з боку тестованого знайти відповіді на питання за допомогою сторонніх програмам або в Інтернеті.

Створення знімків робочої області екрану комп’ютера та фотознімків за допомогою веб камери ПК. В процесі здачі КТ проктор може увімкнути функцію автоматичного створення знімку екрану при зміні розмірів активного вікна програми, та в разі необхідності знімки можна створювати по команді з серверу.

Розпізнавання обличчя тестованого та стеження за його положенням відносно веб-камери. Дана функція в режимі реального часу детектує особу студента, щоб запобігти можливій заміні в процесі іншою людиною. Та стежити за тим щоб студент не відволікався від процесу проходження тесту.

Серверна частина системи моніторингу за е-тестуванням повинна бути розміщена на окремій машині, без доступу до неї сторонніх особі, для роботи з клієнтами їй необхідне хороше з'єднання з мережею Internet, або налаштована локальна мережа.

Основні задачі які повинна виконувати серверна частина розробленого застосунку:

Адаптивна система налаштувань кожного елементу контролю включає в себе повністю варіативну зміну інтервалу передачі даних від клієнта на сервер і відправку повідомлень від серверу на підконтрольні комп’ютери. Точне налаштування інтервалів передачі даних дозволяє проктору більш точно налаштувати клієнтську програму в залежності від ситуації та важливості тесту. Функція відправки повідомлень з серверу застосовується в ситуаціях, коли проктору необхідно надіслати повідомлення або попередження конкретному студенту.

Забезпечення роботи серверу на пристроях з невеликою з невеликою роздільною здатністю екрану. Забезпечення такої можливості в подальшому може дати змогу екзаменатору виконувати налаштування та стежити за процесом тестування з будь-якого пристрою в будь-якій частині світу.

Користувацький інтерфейс серверної частини розділяє перегляд результатів та можливість налаштування клієнтських застосунків. Для цього функціонал веб-застосунку був розділений між декількома сторінками. Можливий вигляд макету сторінки налаштувань та перегляду повідомлень від клієнтських пристроїв наведено на рисунку 3.1.

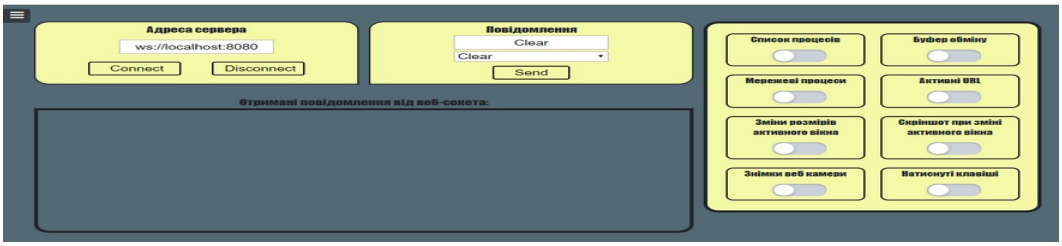

Рисунок 3.1 – Макет сторінки веб-застосунку на етапі розробки

Функція контролю отриманих від контрольованих машин, за час спостережень, включає в себе прийом, зберігання та відображення в зручному форматі даних. Отримані в процесі тестування результати спочатку зберігаються в БД на комп’ютері тестованого, а після закінчення тестування зібрана інформація автоматично зберігається на віддаленому сервері.

Відображення результатів виконується вибором необхідної таблиці зі списку. Відображати можна не тільки поточні результати, але також і результати попередніх «сесій тестування». В разі необхідності також можлива фільтрація виводу за допомогою вибору необхідний рядків таблиці.

Вивантаження інформації в окремий файл. Можливість що дозволяє, в разі необхідності більш детального ознайомлення, створити спеціальний файл, з даними конкретної таблиці або всієї БД, в форматі .csv, для подальшого ознайомлення студентом, або викладачем.

3.3 Архітектура клієнтської частини

Програмне рішення, що створене в Visual Studio, складається з десяти проектів (рисунок 3.2).

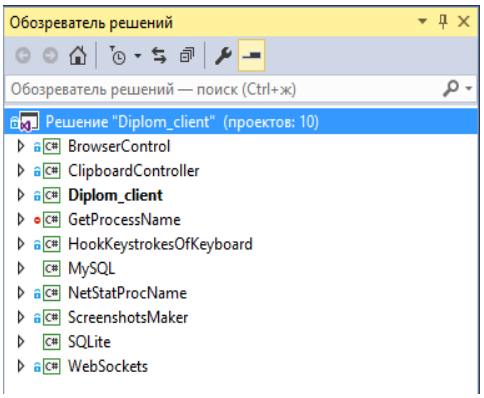

Рисунок 3.2 – Файлова структура проекту

У випадку з використанням декількох проектів поділ вихідного коду реалізується за допомогою поділу функціоналу програмного рішення на окремі складові, які реалізуються в різних за призначенням проектах. Це дозволяє розбити функціонал клієнтського застосунку на окремі функціональні блоки. Такий підхід спрощує, прискорює роботу над кожним окремим елементом, та дозволяє спростити процес оптимізації кожної окремої частини, а отже і всього проекту в цілому.

Клієнтська частина програми складається з таких основних частин.

Проект Browser Control – реалізує функціонал спостереження та виявлення активного URL в браузерах.

Проект Clipboard Controller – стеження за змінами буфера.

Проект GetProcessName – отримання списку відкритих на ПК процесів.

Проект HookKeystrokesOfKeyboard реалізує функціонал обробки та логування натиснутих користувачем клавіш на клавіатурі.

Проект Mysql – призначений для роботи та синхронізації даних з серверною базою даних.

Проект NetStatProcName – отримання в режимі реального часу процесів на ПК користувача.

Проект Screenhotsmaker – в даному проекті описується функціонал, призначений для роботи з зображеннями, а саме створення скріншотів всієї області екрану комп'ютера та знімків за допомогою веб-камери, при зміні активного вікна програми або його розмірів.

Проект Sqlite – описує роботу та взаємодію програми з клієнтською базою даних.

Проект WebSockets – в цьому проекті описуються способи взаємодії клієнтів з серверною частиною застосунку.

Проект Diplom_client – виконує в програмі функцію головного проекту, у якому описується інтерфейс користувача і основні правила роботи інших проектів.

3.4 Архітектура серверної частини

Файлова структура серверної частини застосунку відповідає стандартній структурі застосунків, що розробляються з використанням можливостей CMS WordPress. Файлова структура вихідного коду веб-застосунку складається з великої кількості стандартних файлів самої CMS, та файлів розробленої «теми», в якій описані функціональні можливості

програми, включаючи інтерфейс користувача, логіку доступу та обробки даних, роботу з користувачами і т.д.

Для роботи CMS WordPress використовує трирівневу архітектуру MVC (рисунок 3.3) через те, що це найбільш широко використовуваний та один з найбільш простих та надійних шаблонів проектування. Дана архітектура не накладає жодних обмежень на спосіб реалізації інтерфейсу користувача та забезпечує поділ відповідальності між функціональними блоками.

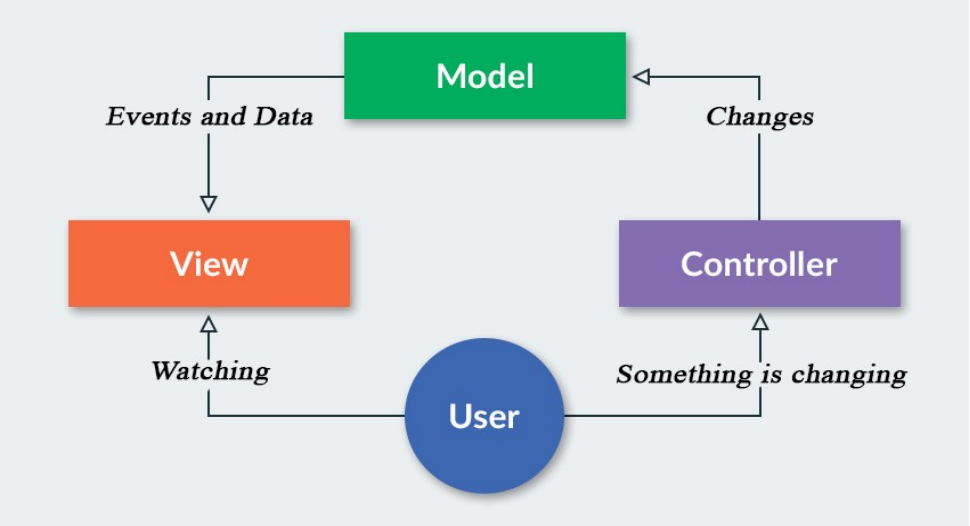

Рисунок 3.3 – Трирівнева архітектура шаблону MVC в проекті

Трирівнева архітектура характеризується тим, що це дуже проста в використанні схема, завдяки якій відкривається можливість реалізувати проекти зі складною внутрішньою логікою.

Веб-застосунок (серверна частина програми) містить в собі три основні сторінки, з якими може працювати користувач.

На головній сторінки відображається інформація про всі дії, які відбуваються на клієнтських машинах, та є можливість відправляти повідомлення клієнтам в процесі тестування.

Сторінка роботи з текстовими даними дозволяє ознайомитися з текстовими результати, які були зібрані за час спостереження.

Сторінка роботи з зображеннями дозволяє користувачеві ознайомитися з фотознімками та скріншотами, які були зроблені під час тестування.

3.5 Реалізація клієнтської частини програми

Для роботи з різними браузерами використовуються класи Chorme, Firefox, Opera, Microsoft Edge, кожен з яких реалізує функціонал отримання активного URL. Для визначення запущених браузерів використовуються спеціальні поля (приклад 3.1), в залежності від вмісту програма буде викликати необхідний клас, це зв'язано з неможливістю використання одного підходу через різний підхід в побудові елементів в браузерах.

```
      Process[] procsChrome = Process.GetProcessesByName("chrome");
      Process[] process = Process.GetProcessesByName("firefox");
      Process[]  procsOpera  =  Process.GetProcessesByName("opera");  return
edgeWindow.FindFirst(TreeScope.Children,
      new                  AndCondition(new                  PropertyCondition
AutomationElement.ControlTypeProperty , ControlType.Window ),
      new   PropertyCondition(   AutomationElement.NameProperty,   "Microsoft
Edge" )))
```

Приклад 3.1 – Поля для визначення імені використовуваного браузера

Однією з основних особливостей програми є можливість роботи з віддаленими базами даних. Для цього в проекті реалізована функція підключення (приклад 3.2), в яку передаються налаштування після встановлення зв’язку з сервером веб-застосунку.

```
      public static string datasource= ""; public static string port= "";
public static string username= ""; public static string password= ""; public
static string database= ""; public string check_connection = ""; public void
ConnectionToMySQL(){ Connection = @"datasource=" + datasource +";port=" +
port + ";username=" + username +";password=" + password + ";database=" +
database; try{Con_to_db = new MySqlConnection(Connection);Con_to_db.Open();
Comand_for_db     =     new     MySqlCommand("SELECT     1",     Con_to_db);
Comand_for_db.CommandTimeout    =    60;    check_connection_logic    =    true;
check_connection     =     "Connected";}     catch     (MySqlException     ex)
{ check_connection_logic = false;check_connection = "Disconnected";}}
```

Приклад 3.2 – Функція підключення до віддалених БД

Для запобігання втрати даних використовується система синхронізації даних між клієнтом і сервером (приклад 3.3), принцип якої полягає в автоматичній синхронізації тільки використовуваної в даний момент часу таблиці.

```
      Comand_for_db.CommandText = "INSERT INTO Buffer (buff,Time,Ids) values
(@buf,@Time,@Ids)"; Con_to_db.Open();
      Mysql_sqlite_result_buffer = sqlite.Table_buffer.Clone();
      sqlite.Table_buffer.AsEnumerable().Except(Mysql_table_buffer.AsEnumerabl
e(),              DataRowComparer.Default).ToList().ForEach(r              =>
Mysql_sqlite_result_buffer.ImportRow(r)); if (sqlite.Table_buffer != null &
Mysql_table_buffer     !=     null)     {foreach     (DataRow     row     in
Mysql_sqlite_result_buffer.Rows)
{Comand_for_db.Parameters.AddWithValue("@buf",                          row[0]);
Comand_for_db.Parameters.AddWithValue("@Time",                          row[1]);
Comand_for_db.Parameters.AddWithValue("@Ids",                           row[2]);
Comand_for_db.ExecuteNonQuery();     Comand_for_db.Parameters.Clear();}     }
CanAddToServer1 = true; Con_to_db.Close(); }
```

Приклад 3.3 – Приклад синхронізації даних для таблиці Buffer

Однією з основних функцій програми є створення фотознімків (приклад 3.4) та зображень всієї області екрану комп'ютера.

```
      Bitmap  bmpScreenshot  =  new  Bitmap(Screen.PrimaryScreen.Bounds.Width,
Screen.PrimaryScreen.Bounds.Height, PixelFormat.Format32bppArgb);
      public void ScreeshotMaker() {
                  var gfxScreenshot = Graphics.FromImage(bmpScreenshot);
                   gfxScreenshot.CopyFromScreen(Screen.PrimaryScreen.Bounds.X,
Screen.PrimaryScreen.Bounds.Y,    0,    0,    Screen.PrimaryScreen.Bounds.Size,
CopyPixelOperation.SourceCopy);}
```

Приклад 3.4 – Створення знімку робочої області екрану

В момент відкриття програми генерується набір параметрів для підключення до серверу та унікальний ідентифікатор користувача, завдяки якому можлива робота та обробка даних кожного окремого користувача програми (приклад 3.5). Робота в різних режимах стала доступною завдяки розробленій спеціальній системі команд, в залежності від яких може змінюватися функціонал програми.

```
      public void Websock_Client_Node(Object source, ElapsedEventArgs el){
      try{if       (!web_socet_work){       WebSocketClient       =       new
WebSocketSharp.WebSocket(serverUri.ToString());  WebSocketClient.OnMessage  +=
wsServer_NewDataReceived;              WebSocketClient.OnOpen              +=
wsServer_NewSessionConnected;  WebSocketClient.OnError  +=  wsServer_OnError;
WebSocketClient.OnClose += wsServer_Disconnect; identificator.GetRandom(); id
=    IdOfUser.numer;    WebSocketClient.Connect();}}catch    (Exception    ex)
{ web_socet_work = false;}}
```

Приклад 3.5 – Ініціалізація змінних для підключення до серверу

Для роботи з EMGU.CV, нам знадобляться як стандартні типи змінних (int , string ,Image, List, MCvFont) так і спеціалізовані (Capture, HaarCascade, FilterInfoCollection, VideoCaptureDevice) основною задачею яких і є робота з відео потоком та детектування облич за допомогою відео камери, та збереження результатів навчання моделі. Основною ж змінною в проекті є HaarCascade face, яка буде використовуватися в більшій частині проекту, адже саме в ній знаходиться інформація про початкову навчальну вибірку, яка в свою чергу зберігається в файлі haarcascade_frontalface_default.xml

3.6 Реалізація серверної частини програми

Основною функцією серверного застосунку є можливість відправки повідомлень (команд) підконтрольним машинам, для цього був розроблений механізм ініціалізації серверу (приклад 3.6).

```
      const fs = require('fs'); const http = require('http');
      const Websocket = require('websocket').server;
      const index = fs.readFileSync('./mainpage.php', 'utf8');
      const server = http.createServer((req, res) => {
        res.writeHead(200);  res.end(index);}); server.listen(8080, () => {
        console.log('Listen port 8080');});
const ws = new Websocket({httpServer: server, autoAcceptConnections: false});
      const  clients  =  [];ws.on('request',  req  =>  {  const  connection  =
req.accept('',  req.origin);  clients.push(connection);  console.log('Connected
' + connection.remoteAddress); connection.on('message', onWsMessage);
 function onWsMessage(message) {const dataName = message.type + 'Data'; const
data = message[dataName];
clients.forEach(client=>{if(connection !== client) {client.send(data}});}});
```

Приклад 3.6 – Ініціалізація серверу

Ініціалізація проходить в декілька етапів: підключення необхідних модулів, після чого ініціалізуються сторінки, на які будуть відображатися отримані повідомлення, після чого починається прослуховування всіх з'єднань і вхідних повідомлень за встановленим портом (приклад 3.7).

```
<?php $Comands = carbon_get_theme_option('all_comands');?>
<?php foreach ( $Comands as $comand ){ ?>
<div><p><?php echo $comand['comand'];?></p>
```

Приклад 3.7 – Створення списку команд в головному вікні програми

Однією з особливостей серверної частини застосунку є можливість під час, або після тестування вивантажити зібрані дані в форматі csv. Сам процес створення файлу складається з таких етапів: користувач обирає зі списку доступних таблиць ті, з яких необхідно отримати результати, далі перевіряється наявність обраних таблиць, після чого отримується їх вміст і створюється новий файл в форматі csv (приклад 3.8).

```
date_default_timezone_set('Europe/Kiev');
$today=date('d.F.Y~G.i.s');
$new_csv=fopen($_SERVER['DOCUMENT_ROOT'].
"/Diplom_wp/wp-content/themes/diplom/files_CSV/$today.csv", 'w');
fputcsv($new_csv, $bufer, ",", '"' );
}}
```

Приклад 3.8 – Створення файлу в форматі csv

4 ОПИС ПРОГРАМИ ЇЇ ТЕСТУВАННЯ І ІНСТРУКЦІЯ КОРИСТУВАЧА

4.1 Детектування порушень системою

Для початку детектування системою порушень в режимі реального часу, необхідно спочатку становити на ПК користувача приховану програму «клієнт». Перед початком тестування з серверу відправляється команда про увімкнення спостереження (рисунок 4.1).

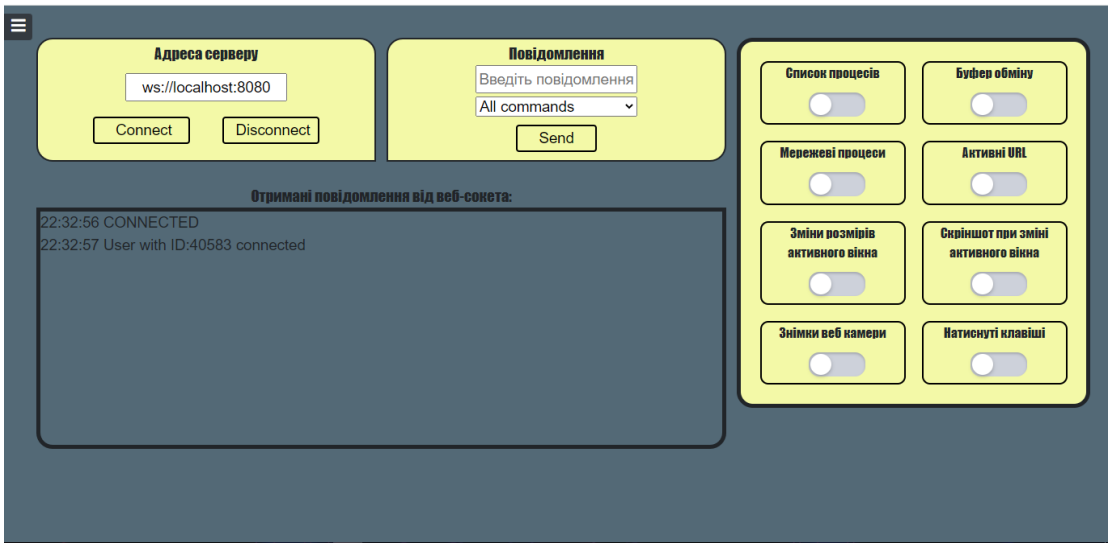

Рисунок 4.1 – Старт нової сесії

В свою чергу користувач не може взаємодіяти з інтерфейсом клієнтського застосунку, і тому не знає що за ним спостерігають. В цей час з серверу на клієнтські застосунки відправляється команда, «стеження за натиснутими клавішами».

Через деякий час, приблизно 5 секунд на сервер починають приходити повідомлення про детектування порушень з боку студентів. Проктор може бачити ідентифікатор користувача, час детектування та тип порушення. Системою було помічено те що користувач в процесі тестування користується клавіатурою, що є недопустимим.

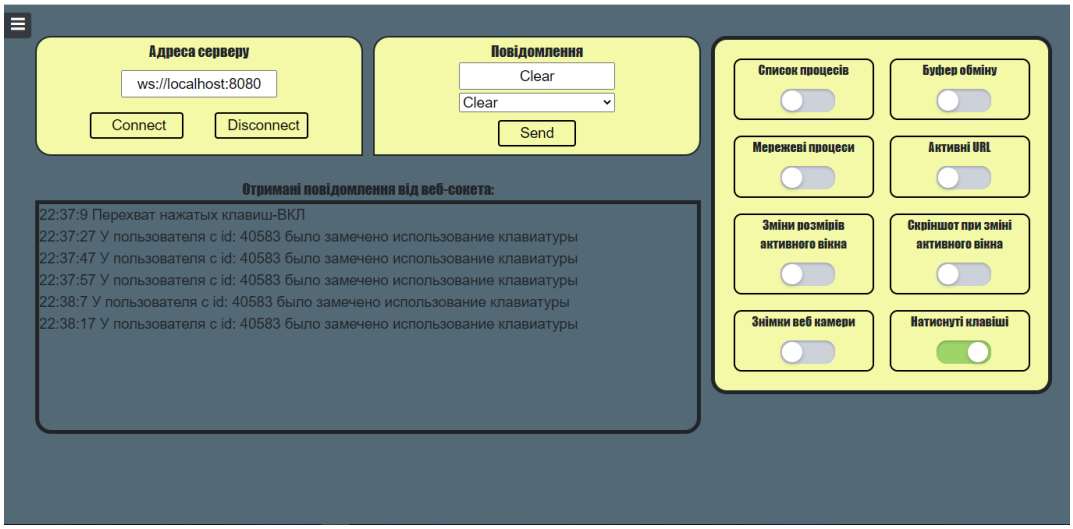

Рисунок 4.2 – Детектування порушень в режимі реального часу

Для того щоб відправити попередження конкретному студенту, проктору необхідно просто ввести ідентифікатор користувача і повідомлення в поле блоку «Повідомлення», та натиснути кнопку «Send». Після чого студент з введеним ідентифікатором отримає повідомлення у вигляді вспливаючого вікна.

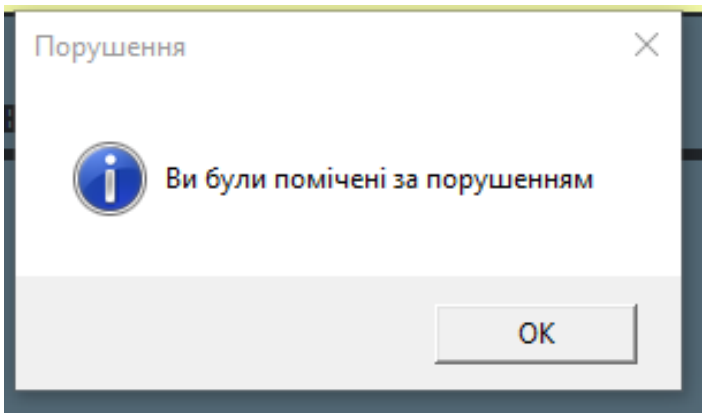

Рисунок 4.3 – Попередження про порушення

Після отримання такого повідомлення можна продовжити спостереження або просто завершити тест цьому студенту. В свою чергу такий підхід дозволяє спочатку просто попередити користувача про

задетектоване порушення без необхідності закривати тестування.

4.2 Тестування програми

Тестування програми проводилося на двох різних комп'ютерах з різними версіями Visual Studio, а саме в Visual Studio 2017, Visual Studio 2015 та Visual Studio 2010 де була перевірена основна функціональність програми і проведена перевірка на сумісність з іншими версіями компілятора Visual Studio. Тестування модулів програми показало, що окремі функції цих модулів повністю працездатні і можуть працювати повністю автономно.

Для моніторингу за користувачами використовується, повністю схований від системних засобів компьютера, програмний застосунок, який автоматично підключаєтсья до віддаленго сервера системи спостережння, та синхронізує всі налаштування, а саме:

– інтервал отримання активних URL на комп’ютері тестованого;

– увімкнення або вимкнення необхідності відстеження буд-яких змін параметрів (ширини та висоти) активного вікна;

– проміжки часу зчитування списку активних процесів;

– отримання переліку мережевих підключень;

– автоматичне зчитування місту буфера обміну та ведення логу натиснутих клавіш;

– інтервальне створення фотознімків за допомогою камери ПК та скріншотів робочої області екрану;

– розпізнавання обличчя тестованого та стеження за його положенням;

В результаті експериментів була побудована таблиця, в якій наглядно видно найбільш поширені способи списування, якими можуть скористатися студенти, загальна кількість детектувань кожного з порушень, і які з порушень вдалося помітити програмі в режимі автоматичного сканування, та за допомогою проктора.

4.3 Хід експериментів

В ході проведення експериментів, була змодельована ситуація, в ході якої користувач, в нашому випадку студент, будучи на дистанційному навчанні, самостійно встановлює розроблений застосунок на свою робочу машину та починає проходити будь-який електронний тест намагаючись максимально непомітно для системи списати.

Так як експерименти в більшості випадків проводилися в робочий час, то це максимально наблизило процес до реальної ситуації проходження КТ, коли студент повинен бути максимально уважним та додатково робити спроби списати відповіді на незнайомі йому питання.

В першому випадку, моніторинг проводився в повністю автономному режимі (без спостереження проктора). В цей час на сервері були увімкнено максимальний режим стеження, який включає в себе задіяння усіх доступних функцій застосунку з можливістю запису порушень в режимі реального часу. Результати спостереження системи та детектування порушень представлено в таблиці 4.1.

Додатково було проведено ще одну сесію спостережень в якій був задіяний проктор. В обов'язки проктора входило, налаштування максимального рівня спостереження, стеження за повідомленнями від клієнтських застосунків, відправка попереджень студентам і ті. Результати спостереження з допомогою проктора представлено в таблиці 4.2.

4.4 Результати експериментів

Завдяки карантинним обмеженням тестування програми вдалося виконати на більше ніж 100 робочих машин студентів.

Основною задачею експерименту було визначити, чи можливо в процесі проходження е-тестування, непомітно для систем та проктора скористатися «шпаргалками».

Таблиця 4.1 – Результати експериментів (автоматичне спостереження)

№	Порушення	Кількість студентів	Кількість помічених порушень системою
1	Нове вікно браузеру	35	35
2	Нова вкладка браузеру	35	35
3	Зміна розмірів головного вікна	17	17
4	Увімкнення сторонніх утиліт	8	7
5	Підключення через віддалений робочий стіл	3	1
6	Пошук відповіді за запитом в Інтернеті	45	45
7	Використання другого монітору	13	13
8	Використання телефону	7	7
9	Допомого сторонньої особи	3	3
10	Поворот голови від екрану	9	8
11	Переключення між різними програми або браузерами	15	15
12	Вимкнення системи моніторингу	1	1

Таблиця 4.2 – Результати експериментів (з проктором)

№	Порушення	Кількість студентів	Кількість помічених порушень системою
1	Нове вікно браузеру	35	35
2	Нова вкладка браузеру	40	35
3	Зміна розмірів головного вікна	19	19
4	Увімкнення сторонніх утиліт	4	2
5	Підключення через віддалений робочий стіл	1	1
6	Пошук відповіді за запитом в Інтернеті	32	32
7	Використання другого монітору	13	13
8	Використання телефону	9	9
9	Допомого сторонньої особи	5	5
10	Поворот голови від екрану	11	10
11	Переключення між різними програми або браузерами	21	21
12	Вимкнення системи моніторингу	1	1

Додатково було проведено експеримент, в якому розроблений застосунок зрівнюється з декількома популярними аналогами, а саме AnyDesk, Chrome Remote Desktop, Віддалений доступ RMS, Remote Utilities, Microsoft Remote Desktop, Exam Monitor. Для проведення експерименту була створена ситуація максимально наближена до справжнього тестування, тому для контролю за якістю тестування був задіяний проктор. Результати експерименту приведені в таблиці 4.3

Таблиця 4.3 – Порівняльний аналіз застосунку з аналогами

№	Порушення	Кількість студентів	CRD	MRD	AnyDesk	Кількість помічених порушень
1	Нове вікно браузеру	15	15	15	15	15
2	Нова вкладка браузеру	40	30	32	38	35
3	Зміна розмірів головного вікна	19	0	0	0	19
4	Увімкнення сторонніх утиліт	3	0	0	0	2
5	Підключення через віддалений робочий стіл	2	0	0	0	1
6	Пошук відповіді за запитом в Інтернеті	39	37	35	35	39
7	Використання другого монітору	13	13	12	12	13
8	Використання телефону	9	0	0	0	9
9	Допомого сторонньої особи	5	0	0	0	5
10	Поворот голови від екрану	11	0	0	0	10
11	Переключення між різними програми або браузерами	21	20	18	20	21
12	Вимкнення системи моніторингу	1	1	1	1	1

Після проведених експериментів, маємо такі результати роботи. Найкращий результат детектування порушень показав саме автоматичний режим роботи та стеження (Таблиця 4.1). В такому режимі система задетектувала (187/191). На відміну від контролю проктором, такий підхід дає можливість повністю автоматизувати процес, коли є необхідність проведення швидкого тестування, без можливості звершити тест порушнику. Однак при безпосередній участі проктора (Таблиця 4.2), кількість задетектованих порушень склала (183/191), однак реагування на порушення було більш жорстким, а саме відключення тестованого від системи е-тесту.

Також, після проведення додаткових, тестів, де зрівнювалася працеспроможність аналогів, та розробленого застосунку (Таблиця 4.3), показала повну готовність розробленої системи до неочевидних спроб та методів списування загальна кількість задетектованих помилок дорівнює (160/178).

В свою чергу аналоги, показали в цьому випадку свою недостатню укомплектованість та неможливість відстежувати велику кількість порушень. CRD (116/178), MRD(113/178), AnyDesk(121/178).

Для оцінки ефективності розробленого застоснку в порівнянні з аналогами використовується сукупність критеріїв оцінки ефективності, що дозволяють визначити реакцію детектування порушень. Основним критерієм завдяки якому досягається високий рівень точності детектування, є час детектування порушень. На відміну від аналогів, в розробленій системі є можливість налаштування часових інтервалів для функцій детектування: час визначення активного URL в різних типах браузерів, час відстеження змін вмісту буфера обміну, час отримання списку активних процесів на комп'ютері користувача, час обробки та логування натиснутих користувачем клавіш на клавіатурі, час отримання списку мережевих процесі, час створення скріншотів всієї області екрану комп'ютера та знімків за допомогою веб-камери, при зміні активного вікна програми або його розмірів.

На основі даних з таблиць 4.1 – 4.3 можна зробити зрівнювальний аналіз розробленої системи та аналогів. До основних типів порушень, які системи найгірше помічали можна віднести: Зміна розмірів головного вікна, Увімкнення сторонніх утиліт, Підключення через віддалений робочий стіл, Використання телефону, Допомога сторонньої особи, Поворот голови від екрану. В таблиці 4.4 наглядно показано як впоралися аналоги в порівнянні з розробленим застосунком.

Таблиця 4.4 – Порівняльний аналіз по найбільш складно детектованим порушенням

№	Порушення	Кількість порушень	CRD	MRD	AnyDesk	Розроблений застосунок
1	Зміна розмірів головного вікна	19	0/19	0/19	0/19	19/19
2	Увімкнення сторонніх утиліт	3	0/3	0/3	0/3	2/3
3	Підключення через віддалений робочий стіл	2	0/2	0/2	0/2	1/2
4	Використання телефону	9	0/9	0/9	0/9	9/9
5	Допомого сторонньої особи	5	0/5	0/5	0/5	5/5
6	Поворот голови від екрану	11	0/11	0/11	0/11	10/11

Як видно за таблиці 4.4 завдяки високому критерію ефективності детектування порушень, розроблений застосунок детектує більшу частину порушень, та може біль точно виявити тип самого порушення.

4.5 Клієнтська частина програми

Після запуску клієнтської частини програма повністю приховується від очей користувача та від інших програм детектування.

В разі необхідності користувач може відкрити вікно програми, використавши для цього зарезервовану команду «Shisft + H». А у разі необхідності сховати вікно від очей користувача за стосунок можна знову сховати за допомогою спеціальної команди «Shisft + S».

Також в разі виникнення раптового відключення застосунку від серверу, повторне підключення буде автоматично проводитися через кожні 5 секунд.

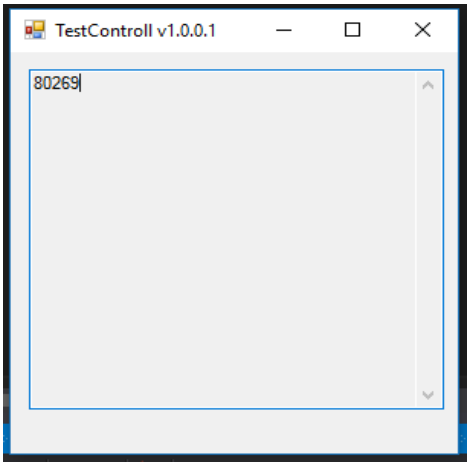

Рисунок 4.4 – Головне вікно клієнтської частини застосунку

4.6 Авторизація на сайті

Для запобігання спроб несанкціонованого доступу до можливостей серверу, користувач повинен пройти обов'язкову процедуру авторизації.

Головна сторінка, містить в собі всі необхідні для моніторингу за процесом тестування функції: Налаштування підключення до клієнтський машин, відображення всіх повідомлень про порушення від клієнтів. Сторінка ознайомлення з текстовими результатами, використовується для перегляду тестових даних про порушення, та коли воно було здійснено. Для перегляду зроблених знімків веб-камери та скрінотів робочого столу, використовується сторінка «Отримані знімки» на якій користувач може переглянути всі зроблені в процесі спостереження за тестуванням.

4.7 Взаємодія з клієнтськими застосунками

Головна сторінка сайту (рисунок 4.5) необхідна для стартового налаштування всіх необхідних елементів системи пошуку та детектування. Користувач з рівнем доступу «адміністратор» та «редактор» бачить основні функціональні блоки, які використовуються напряму для забезпечення взаємодії з клієнтськими застосунками, для обробки вхідних повідомлень.

Блок «Адреса серверу» – необхідний для ініціалізації та втановлення нового з'єднання з сервером. Для цього використовується протокол WebSocket.

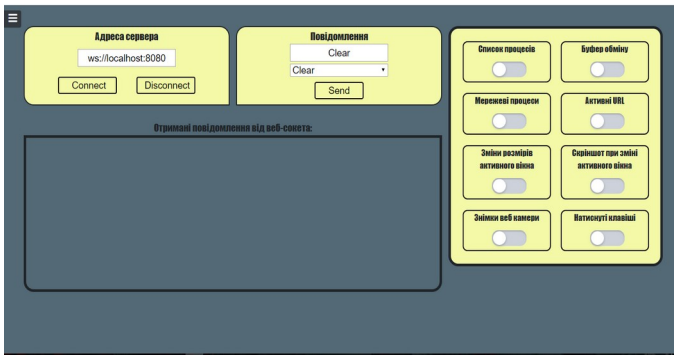

Рисунок 4.6 – Сторінка налаштувань серверу

Блок «Повідомлення» використовується напряму для віпризначений для відправки повідомлень та команд для зміни налаштувань на підконтрольних машинах або у конкретного користувача (рисунок 4.7).

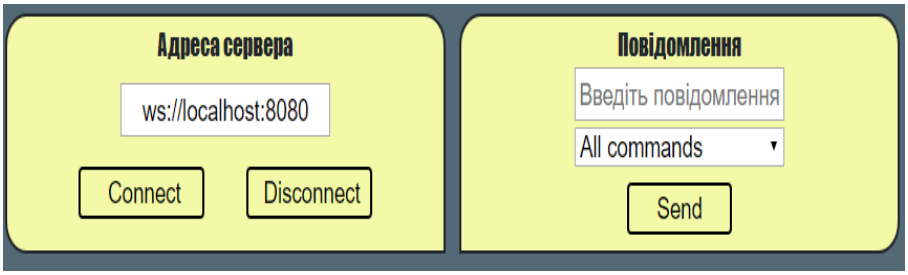

Рисунок 4.7 – Блок «Адреса сервера» та блок «Повідомлення»

Блок з доступними функціями клієнтських застосунків знаходиться в правому «сайдбарі» і використовується для більш чіткого налаштування функціоналу клієнтських застосунків (рисунок 4.8).

Кожен з перемикачів призначений для відправки конкретної команди всім підключеним користувачам, і його стан визначає поточний активний функціонал клієнтських застосунків (рисунок 4.8).

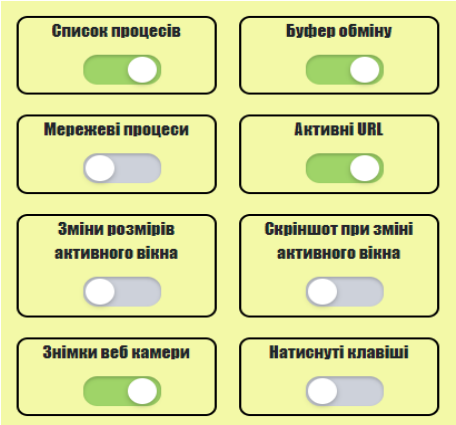

Рисунок 4.8 – Приклад активного функціоналу

Блок «Отримані повідомлення від веб-сокета» використовується для відображення всіх отриманих повідомлень від підконтрольних машин (рисунок 4.9).

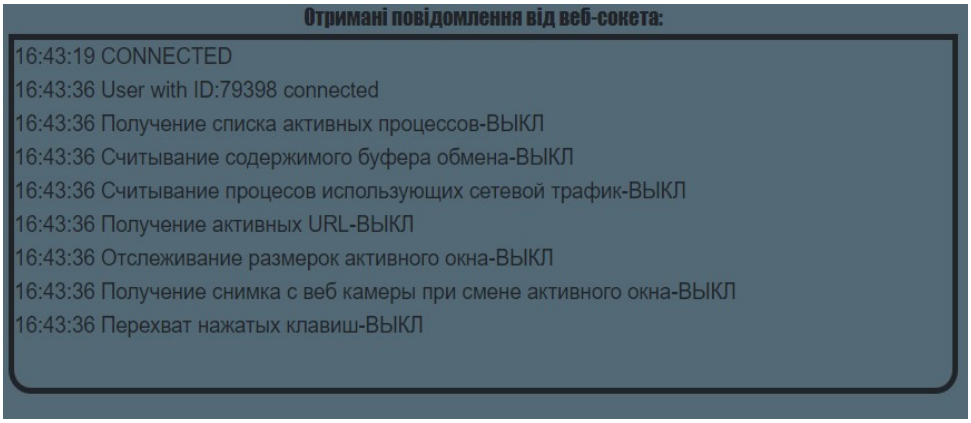

Рисунок 4.9 – Приклад отриманих повідомлень

4.8 Обробка результатів моніторингу

На сторінці розташовані основні блоки для відображення та обробки інформації, що зібрана за час спостереження (рисунок 4.10).

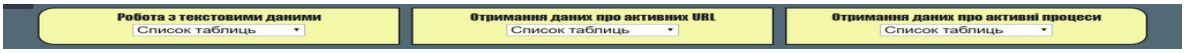

Рисунок 4.10 – Основні блоки сторінки «Текстові результати»

Блок з назвою «Робота з текстовими даними» використовується для отримання зібраних результатів про вміст буферу обміну та для отримання списку натиснутих клавіш клавіатури. Блок з назвою «Отримання даних про активні URL» використовується для отримання списку поточних активних URL браузера та для отримання всіх змін параметрів активного вікна. Блок з назвою «Отримання даних про активні процеси» використовується для отримання списку активних процесів і мережевих підключень (рисунок 4.11).

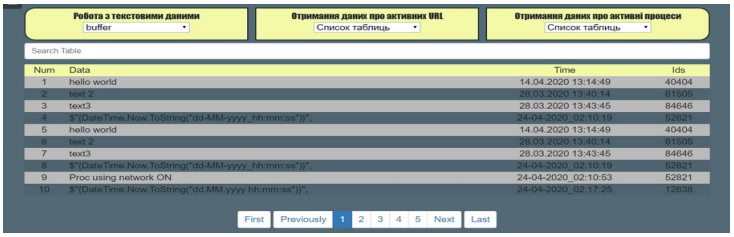

Рисунок 4.11 – Основні елементи сторінки «Текстові результати»

Для більш зручної роботи з отриманими даними, загальна таблиця розбивається на необхідну кількість сторінок в залежності від загального об'єму отриманої інформації.

В разі необхідності отримувати дані тільки за поточний період часу, в головному меню знаходиться перемикач, завдяки якому є можливість змінювати загальний час, за який будуть відображені зібрані дані

(рисунок 4.12).

Після отримання необхідної інформації з таблиці, користувач може скористатися вбудованою пошуковою системою, для отримання більш докладної інформації про заданий період часу, діяльність конкретного студента, пошук по конкретним словом і т.д. (рисунок 4.12).

Рисунок 4.12 – Приклад використання пошуку

Для роботи з зібраними за час спостереження знімками, використовується сторінка «Отримані знімки». Кнопка «Фотографії» використовується для отримання зібраних фотографій зроблених за допомогою веб-камери. Для відображення отриманих знімків робочої області екрану комп'ютера на клієнтських машинах, використовується кнопка «Screens» (рисунок 4.13).

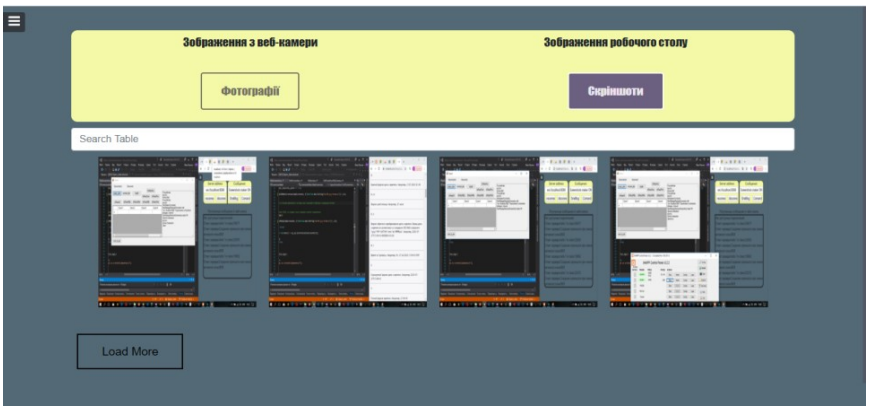

Рисунок 4.13 – Основні елементи сторінки «Отримані знімки»

Після отримання списку зображень, користувач може подивитися зображення в більш зручній формі, або в форматі «слайдшоу» (рисунок 4.14).

У разі необхідності отримання зібраних даних для більш глибокого аналізу, на сайті передбачена можливість вивантаження всіх зібраних даних в окремому файлі в форматі csv (рисунок 4.15).

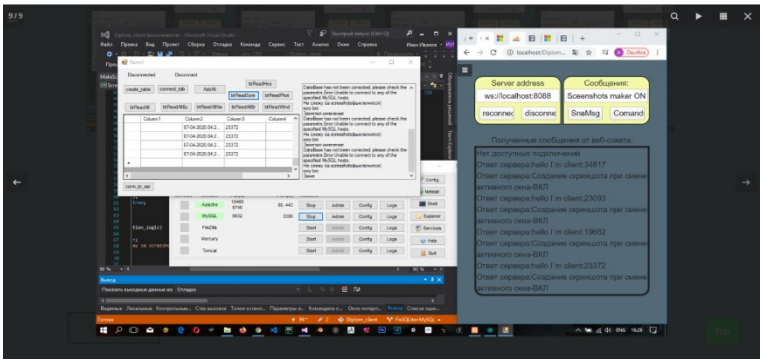

Рисунок 4.14 – Відображення скріншотів в форматі «слайдшоу»

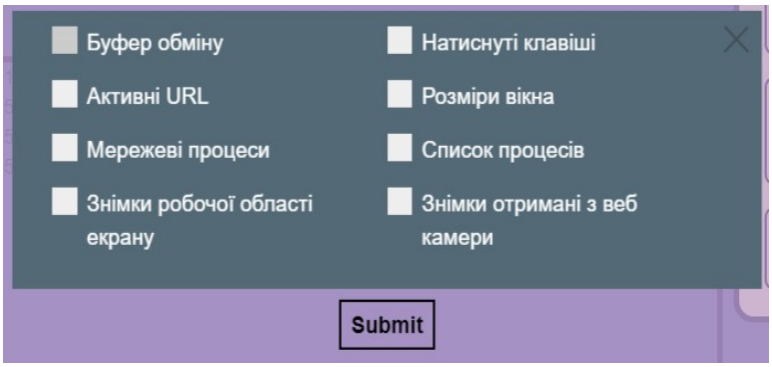

Рисунок 4.15 – Список таблиць доступних для отримання результатів

Для цього в верхній частині меню передбачена кнопка, при натисканні на яку відкривається окреме модальне вікно, в якому відображається список всіх доступних таблиць, з яких є можливість вивантажити дані

ВИСНОВКИ

Поставлені в роботі задачі виконані в повному обсязі. Розроблена система спостереження за поведінкою студентів дозволяє в режимі реального часу відстежувати порушення під час проведення е-тестування. Об'єктом дослідження є системи моніторингу за е-тестуванням. Предмет дослідження: використання систем моніторингу як спосіб стеження якості проведення е-тестування.

У ході виконання кваліфікаційної роботи був проведений аналіз великої кількості існуючих програмних рішень, як спеціалізованих так і програмних. в ході дослідження були виявлені основні переваги та недоліки даних застосунків. На основі проведеного аналізу існуючих рішень були сформульовані головні вимоги до функціоналу та принципи роботи застосунку.

Реалізація клієнтської частини створена з використанням мови програмування C# та платформи .NET. В свою чергу серверна частина за стосунку розроблена з використанням CMS WordPress. Для створення користувацького інтерфейсу та організації взаємодії з ним задіяні можливості HTML, CSS, JavaScript, PHP та C#.

Для подальшого розвитку ПЗ передбачається модернізація системи синхронізації даних між клієнтськими машинами та сервером, більш точне детектування облич в різних положеннях та оптимізація роботи на користувацьких машинах.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Суворов О.О. Програмне забезпечення системи моніторингу рівня поточних знань студентів. URL: https://core.ac.uk/download/81587557.pdf (дата звернення: 03.12.2021)

2. Moodle URL: https://moodle.org/?lang=ru.(дата звернення: 03.12.2021)

3. Застосування комп'ютерного тестування для контролю знань URL: http://vnz.org.ua/statti/3777-zastosuvannja-kompjuternogo-testuvannja-dlja-kontrolju-znan. (дата звернення: 03.12.2021)

4. Комп’ютерне тестування як засіб оцінювання рівня компетентності учнів URL: http://timso.koippo.kr.ua/hmura9/kompyuterne-testuvannya-yak-zasib-otsinyuvannya-rivnya-kompetentnosti-uchniv/. (дата звернення: 03.12.2021)

5. Татарников А.О. Використання технології комп’ютерного зору як методу контролю під час онлайн тестування. *Пріоритетні напрямки та вектори розвитку світової науки:* матеріали I міжнародної студентської наукової конференції., м. Миколаїв, 21 травня 2021. Вінниця. С. 16–17.

6. Татарников А.О. Завдання інтеграції алгоритмів комп'ютерного зору в системах контролю успішності учнів. *Проблеми та перспективи реалізації та впровадження міждисциплінарних наукових досягнень:* матеріали II міжнародної наукової конференції., м. Київ, 27 серпня 2021. Київ. С 117–118.

7. Татарников А.О., Іващенко Г.С. Організація клієнт-серверного взаємодія в системі моніторингу проходження тестування учнів. *Радіоелектроніка та молодь у XXI столітті*: матеріали XXIV міжнародного молодіжного форуму., м. Харків, 7-9 квітня 2020. Харків, 2020. С. 25–26.

8. Архітектура «Клієнт-Сервер» URL: https://itelon.ru/blog/arkhitektura-klient-server/ (дата звернення: 03.12.2021)

9. Экзамус URL: https://hitech.newsru.com/article/12aug2015/examus. (дата звернення: 03.12.2021)

10. Стеження на іспитах: програма ExamCookie URL: https://habr.com/ru/post/453536/. (дата звернення: 03.12.2021)

11. ExamCookie URL: https://www.examcookie.dk/. (дата звернення: 03.12.2021)

12. Забезпечуємо чесність та рівні умови при проведенні онлайн-тестувань URL: https://proctoredu.ru/. (дата звернення: 03.12.2021)

13. Exam Monitor URL: https://sdu.exammonitor.dk/.(дата звернення: 03.12.2021)

14. PROCTORTRACK URL: https://tlt.rutgers.edu/instructional-technology-tools/proctortrack. (дата звернення: 03.12.2021)

15. Кращі програми для віддаленого доступу до комп’ютера URL: https://vistplus.com/stati/krashhi-programi-dlya-viddalenogo-dostupu-do-komp-yutera/. (дата звернення: 03.12.2021)

16. ТОП-4 програм віддаленого доступу до комп'ютера в 2021 році URL: https://ua.softlist.com.ua/articles/top-5-programm-udalennogo-dost/. (дата звернення: 03.12.2021)

17. Програмне забезпечення для моніторингу мобільних телефонів, Tablets та комп'ютерів URL: https://spyera.com/uk/. (дата звернення: 03.12.2021)

18. Віддалений робочий стіл URL: https://www.comss.ru/page.php?id=2887 (дата звернення: 03.12.2021)

19. AnyDesk – віддалене управління комп'ютером і не тільки URL: https://remontka.pro/remote-desktop-anydesk/. (дата звернення: 03.12.2021)

20. Що таке стек технологій і чому це важливо? URL: http://rainbowsoft.ru/technology_stack. (дата звернення: 03.12.2021)

21. Работа с SQLite URL: https://metanit.com/sharp/wpf/21.1.php. (дата звернення: 03.12.2021)

22. Піскунов, О. Г. Неповне керівництво по SQLite для користувачів URL: https://er.nau.edu.ua/bitstream/NAU/10100/6/SQLite.Allow.0.90.pdf. (дата звернення: 03.12.2021)

23. Що являє собою MySQL? URL: http://www.mysql.ru/docs/man/What-is.html. (дата звернення: 03.12.2021)

24. Іващенко Г.С., Татарников А.О. Організація взаємодії та синхронізації даних між sqlite і mysql. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління:*. матеріали десятої міжнародної науково-технічної конференції., м. Харків, 9-10 квітня 2020. Харків. С. 70.

25. К. Нейгел, Б. Ивьен, Д. Глинн, М. Вільямс., C# 5.0 та платформа .NET 4.5 для професіоналів. Київ, 2014. 1440 с.

26. Короткий огляд мови C# URL: https://docs.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/. (дата звернення: 03.12.2021)

27. C# Windows Forms # URL: http://programer.in.ua/index.php/pochatkivtsiu/rozrobka-ihor-dlia-pochatkivtsiv-na-c/200-urok1-c-windows-forms. (дата звернення: 03.12.2021)

28. Windows Forms overview URL: https://docs.microsoft.com/en-us/dotnet/framework/winforms/ windows-forms-overview. (дата звернення: 03.12.2021)

29. Ласкаво просимо в інтегроване середовище розробки Visual Studio URL: https://docs.microsoft.com/ru-ru/visualstudio/get-started/visual-studio-ide?view=vs-2019. (дата звернення: 03.12.2021)

30. Популярні методи виявлення та розпізнавання осіб URL: http://lib.secuteck.ru/articles2/videonabl/populyarnye-metody-obnaruzheniya-i-raspoznavaniya-lits. (дата звернення: 03.12.2021)

31. Навчання каскаду Хаара на прикладі пошуку символів автомобільного номера OpenCV URL: shorturl.at/oGHY2. (дата звернення: 03.12.2021)

32. Татарников А.О. Використання технології комп’ютерного зору при проведенні онлайн тестування. *Інформаційні технології в сучасному світі: дослідження молодих вчених:* матеріали міжнародної науково-

практичної конференції молодих учених, аспірантів та студентів., м. Харків. 18-19 березня 2021. Харків. С. 58.

33. Татарников А.О. Дослідження методів обробки та класифікації емоцій людини при розробці систем комп’ютерного зору. *Теорія і практика сучасної науки*: матеріали II міжнародної науково-теоретичної конференції., м. Краків. 12 листопада 2021. Краків. С. 91–92.

34. Татарников А.О. Використання технології розпізнавання емоцій людини на основі штучних нейронних мереж в освіті. *Цифровізація науки та сучасні тренди її розвитку:* матеріали II міжнародної студентської конференції., м. Миргород 5 листопада 2021. Миргород. С. 86–87.

35. Работа с SQLite URL: https://metanit.com/sharp/wpf/21.1.php(дата звернення: 03.12.2021)

36. SQLite URL: https://ru.bmstu.wiki/SQLite. (дата звернення: 03.12.2021)

37. Застосування бібліотеки AForge.NET і її розширення Accord.NET Framework при розпізнаванні осіб в режимі реального часу URL: https://moluch.ru/archive/154/43602. (дата звернення: 03.12.2021)

38. UI Automation Overview URL: https://docs.microsoft.com/en-us/dotnet/framework/ui-automation/ui-automation-overview. (дата звернення: 03.12.2021)

39. Windows Automation API 3.0 Overview URL: https://www.codemag.com/article/0810042/Windows-Automation-API-3.0-Overview/ (дата звернення: 03.12.2021)

40. Бібліотека Emgu CV та її встановлення URL: http://devnuances.com/c_sharp/emgu-cv-i-e-ustanovka//. (дата звернення: 03.12.2021)

41. Початок роботи з OpenCV та його застосування у C# URL: https://habr.com/ru/post/260741/. (дата звернення: 03.12.2021)

42. Опис CMS WORDPRESS URL: https://stebnev-studio.ru/blog/opisanie-cms-wordpress/#one. (дата звернення:

03.12.2021)

43. ABOUT WP URL: https://www.wp-suspension.com/about-wp/. (дата звернення: 03.12.2021)

44. Коротко про wordpress URL: http://revolweb.ru/site/kratko-o-wordpress. (дата звернення: 03.12.2021)

45. Чим гарний Node.js: практика сучасного веб-програмування URL: https://skillbox.ru/media/code/ chem_khorosh_node_js/. (дата звернення: 03.12.2021)

46. Керівництво по Node.js, частина 1: загальні відомості і початок роботи URL: https://habr.com /ru/company/ruvds/blog/422893/. (дата звернення: 03.12.2021)

47. Що являє собою MySQL? URL: http://www.mysql.ru/docs/man/What-is.html. (дата звернення: 03.12.2021)

48. Як підключитися до MySQL використовуючи ADO.NET URL: https://habr.com/ru/post/169929/. (дата звернення: 03.12.2021)

49. Chapter 1 Introduction to MySQL Connector/NET URL: https://docs.oracle.com/cd/E17952_01/connector-net-en/connector-net-introduction.html/. (дата звернення: 03.12.2021)

50. Система і метод синхронізації бази даних URL: https://patents.google.com/patent/US6226650B1/en. (дата звернення: 03.12.2021)

51. Асинхронний веб, або Що таке веб-сокети URL: https://tproger.ru/translations/what-are-web-sockets/. (дата звернення: 03.12.2021)

52. Татарников А.О. Проблема організації захисту даних при використанні протоколу websocket у клієнт-серверній архітектурі. *Об’єднані наукою: перспективи міждисциплінарних досліджень:* матеріали VII Всеукраїнської науково-практичної конференції студентів, аспірантів та молодих вчених., м. Київ, 12-13 листопада 2020. Київ. С. 223–225.